



RISIKO

Argetim Jusufi - 0000970361
Lorenzo Pellegrino - 0000971455
Andrea Zenere - 0000990124

SOMMARIO

Abstract.....	3
Requisiti.....	4
Raccolta dei requisiti.....	4
Tabella dei requisiti.....	5
Analisi dei requisiti.....	6
Vocabolario.....	6
Diagramma dei casi d'uso.....	8
Scenario: Consulta manuale.....	9
Scenario: Scegli nickname.....	9
Scenario: Accesso gioco.....	10
Scenario: Cerca Partita.....	11
Scenario: Crea Partita.....	11
Scenario: Unione Partita.....	12
Scenario: Gestione lobby.....	13
Scenario: Gestione partita.....	14
Scenario: Scelta combinazione.....	16
Scenario: Disponi Armate.....	17
Scenario: Attacco.....	17
Scenario: Spostamento.....	19
Scenario: Utilizzo chat.....	20
Scenario: Esci partita.....	20
Analisi del Rischio.....	21
Tabella Valutazione dei Beni.....	21
Tabella Minacce/Controlli.....	21
Analisi Tecnologica della Sicurezza.....	22
Security Use Case & Misuse Case.....	23
Security Use Case & Misuse Case Scenari.....	24
Requisiti di Protezione dei Dati.....	27
Tabella dei requisiti.....	27
Analisi.....	28
Analisi Documento dei Requisiti: Analisi delle Funzionalità.....	28
Tabella Funzionalità.....	28
ScegliNickname: Tabella Informazioni/Flusso.....	28
ConsultaManuale: Tabella Informazioni/Flusso.....	28
AccessoGioco: Tabella Informazioni/Flusso.....	29
GestionePartita: Tabella Informazioni/Flusso.....	29
ScritturaLog: Tabella Informazioni/Flusso.....	31
Analisi Documento dei Requisiti: Analisi dei Vincoli.....	32
Tabella Vincoli.....	32
Analisi Documento dei Requisiti: Analisi Interazioni.....	32
Tabella Maschere.....	32
Tabella Sistemi Esterni.....	34

Analisi Ruoli e Responsabilità.....	35
Tabella Ruoli.....	35
Utente: Tabella Ruolo-Informazione.....	35
Utente(Creatore della partita): Tabella Ruolo-Informazione.....	35
Utente(Giocatore): Tabella Ruolo-Informazione.....	36
Scomposizione del Problema.....	36
Tabella Scomposizione Funzionalità.....	36
Tabella Sotto - Funzionalità.....	37
Creazione modello del dominio.....	38
Architetture Logica: Struttura.....	41
Diagramma dei package.....	41
Diagramma delle classi: Dominio.....	42
Diagramma delle classi: ScegliNickname & InterfacciaScegliNickname.....	42
Diagramma delle classi: ConsultaManuale & InterfacciaConsultaManuale.....	42
Diagramma delle classi: AccessoGioco & InterfacciaAccessoGioco.....	43
Diagramma delle classi: GestionePartita & InterfacciaGestionePartita.....	44
Architetture Logica: Interazione.....	45
Diagramma di sequenza: Impostazione del Nickname.....	45
Diagramma di sequenza: Visualizza Manuale.....	45
Diagramma di sequenza: Accesso a una partita tramite Ricerca.....	46
Diagramma di sequenza: Accesso a una Partita tramite Creazione.....	47
Diagramma di sequenza: Accesso a una Partita tramite Unione.....	47
Diagramma di sequenza: Espulsione Giocatore.....	48
Diagramma di sequenza: Avviamento Partita.....	48
Diagramma di sequenza: Scioglimento Lobby.....	49
Diagramma di sequenza: Gioca Partita.....	49
Diagramma di sequenza: Esci dalla partita.....	54
Diagramma di sequenza: Utilizzo Chat.....	54
Architetture Logica: Comportamento.....	55
Diagramma di stato: Stati della Partita.....	55
Diagramma di stato: Stati della Fase di un Turno.....	55
Piano di lavoro.....	55
Piano del collaudo.....	56
Progettazione.....	57
Progettazione Architetturale.....	57
Requisiti non funzionali.....	57
Scelta dell'architettura.....	57
Scelta tecnologica.....	60
Progettazione di dettaglio.....	60
Struttura.....	60
Diagramma di Dettaglio: Dominio-Partita.....	61
Diagramma di Dettaglio: Dominio-Log.....	64
Diagramma di Dettaglio: Interfacce nel Client.....	65
Diagramma di Dettaglio: Interfacce nei Server.....	65

Diagramma di Dettaglio: ScegliNickname.....	66
Diagramma di Dettaglio: ConsultaManuale.....	66
Diagramma di Dettaglio: AccessoGioco.....	67
Diagramma di Dettaglio: GestionePartita.....	68
Diagramma di Dettaglio: Broker.....	69
Diagramma di Dettaglio: Client.....	70
Scegli nickname.....	70
Consulta manuale.....	71
Accesso gioco.....	72
Gestione partita.....	75
React component.....	81
Interazione.....	81
Diagramma di Sequenza: Impostazione del nickname.....	81
Diagramma di Sequenza: Visualizza manuale.....	82
Diagramma di Sequenza: Accesso ad una partita tramite Ricerca.....	82
Diagramma di Sequenza: Accesso ad una partita tramite Creazione.....	83
Diagramma di Sequenza: Espulsione Giocatore.....	84
Diagramma di Sequenza: Avviamento Partita.....	85
Diagramma di Sequenza: Scioglimento Lobby.....	86
Diagramma di Sequenza: Accesso ad una partita tramite Unione.....	87
Diagramma di Sequenza: Gioca Partita.....	88
Diagramma di Sequenza: Esci dalla partita.....	95
Diagramma di Sequenza: Utilizzo Chat.....	95
Comportamento.....	96
Progettazione della persistenza.....	96
Formato File Log.....	96
Progettazione per il collaudo.....	96
Progettazione per il Deployment.....	96
Deployment.....	97
Artefatti.....	97
Deployment Type-Level.....	97

Abstract

L'idea è quella di sviluppare un'applicazione web multigiocatore che realizzi le regole classiche del Risiko con in aggiunta qualche regola, al fine di rendere il gioco più veloce e dinamico senza comunque perdere la sua essenza.

Inoltre, sarà presente una chat che permetta di dialogare con tutti i giocatori contemporaneamente.

Sarà possibile giocare su più mappe oltre a quella classica, ad esempio la mappa di Bologna. La struttura del gioco sarà indipendente dalla mappa.

Requisiti

Raccolta dei requisiti

- 1) L'applicazione deve permettere agli utenti di giocare ad una partita multigiocatore di RISIKO in tempo reale.
- 2) L'applicazione non prevede l'autenticazione degli utenti per partecipare ad una sessione di gioco (*modalità ospite*).
- 3) L'applicazione offre più mappe di gioco, ognuna delle quali è divisa in regioni, a loro volta divise in territori.
- 4) Prima di iniziare la partita i giocatori dovranno essere inseriti in una lobby, che permette una gestione flessibile dei giocatori stessi.
- 5) Vi sono 3 modi perché un utente acceda ad una partita:
 - a) L'utente effettua una ricerca secondo alcuni criteri (numero di giocatori in una partita, mappa della partita, turni massimi giocabili e lingua della chat), quando il sistema trova una corrispondenza tra i vari giocatori la partita verrà poi creata e avviata in automatico. La lobby permetterà al giocatore di abbandonare quest'ultima e di visualizzare gli altri giocatori presenti nella lobby.
 - b) Un utente può creare una partita alla quale è associato un codice. Di tale partita può decidere le opzioni, ovvero può decidere la mappa sulla quale si svolge, il numero di giocatori e i turni massimi giocabili. La lobby sarà gestita dal creatore della partita, che diventerà esso stesso un giocatore e che potrà espellere qualsiasi altro giocatore presente.
 - c) L'utente può inserire il codice di una partita creata da un altro giocatore e si unirà alla lobby gestita da quest'ultimo.
- 6) Il creatore della partita può avviare la partita una volta che tutti i giocatori richiesti sono presenti.
- 7) Il creatore della partita può sciogliere la lobby.
- 8) Una volta che la partita è iniziata, nessun altro utente può accedervi.
- 9) Se un giocatore abbandona la partita in corso il sistema dovrà distribuire equamente i territori e le armate del giocatore uscito.
- 10) Essendoci quattro fasi nel turno di ogni giocatore, il sistema assocerà ad ogni fase del turno un tempo limite entro il quale completarla.
- 11) Un giocatore deve poter comunicare su una chat all'interno della partita.
- 12) Il sistema mette a disposizione un manuale consultabile in ogni momento da parte dell'utente.
- 13) L'applicazione deve essere realizzata con tecnologie web.

Tabella dei requisiti

ID	Requisito	Tipo
R1F	L'applicazione deve permettere agli utenti di giocare ad una partita multigiocatore di RISIKO in tempo reale.	Funzionale
R2F	L'applicazione non prevede l'autenticazione degli utenti per partecipare ad una partita (<i>modalità ospite</i>).	Funzionale
R3F	L'applicazione offre più mappe di gioco, ognuna delle quali è divisa in regioni, a loro volta divise in territori.	Funzionale
R4F	Prima di iniziare la partita i giocatori dovranno essere inseriti in una lobby, che permette una gestione flessibile dei giocatori stessi.	Funzionale
R5F	Per accedere a una partita l'utente può effettuare una ricerca tramite alcune opzioni (numero di giocatori in una partita, mappa della partita, turni massimi giocabili e lingua della chat), quando il sistema trova una corrispondenza tra i vari giocatori la partita verrà poi creata e avviata in automatico. La lobby permetterà al giocatore di abbandonare quest'ultima e di visualizzare gli altri giocatori presenti nella lobby.	Funzionale
R6F	Un utente può creare una partita alla quale è associato un codice. Di tale partita può decidere le opzioni, le stesse citate in R5F . La lobby sarà gestita dal creatore della partita, che diventerà esso stesso un giocatore e che potrà espellere qualsiasi altro giocatore presente.	Funzionale
R7F	L'utente può inserire il codice di una partita creata da un altro giocatore e si unirà alla lobby gestita da quest'ultimo.	Funzionale
R8F	Il creatore della partita può avviare la partita una volta che tutti i giocatori richiesti sono presenti.	Funzionale
R9F	Il creatore della partita può sciogliere la lobby.	Funzionale
R10F	Una volta che la partita è iniziata, nessun altro utente può accedervi.	Funzionale
R11F	Se un giocatore abbandona la partita in corso il sistema dovrà distribuire equamente i territori e le armate del giocatore uscito.	Funzionale
R12F	Essendoci quattro fasi nel turno di ogni giocatore, il sistema assocerà ad ogni fase del turno un tempo limite entro il quale completarla.	Funzionale
R13F	Un giocatore deve poter comunicare su una chat all'interno della partita.	Funzionale
R14F	Il sistema mette a disposizione un manuale consultabile in ogni momento da parte dell'utente.	Funzionale
R1NF	L'applicazione deve essere realizzata con tecnologie web.	Non Funzionale

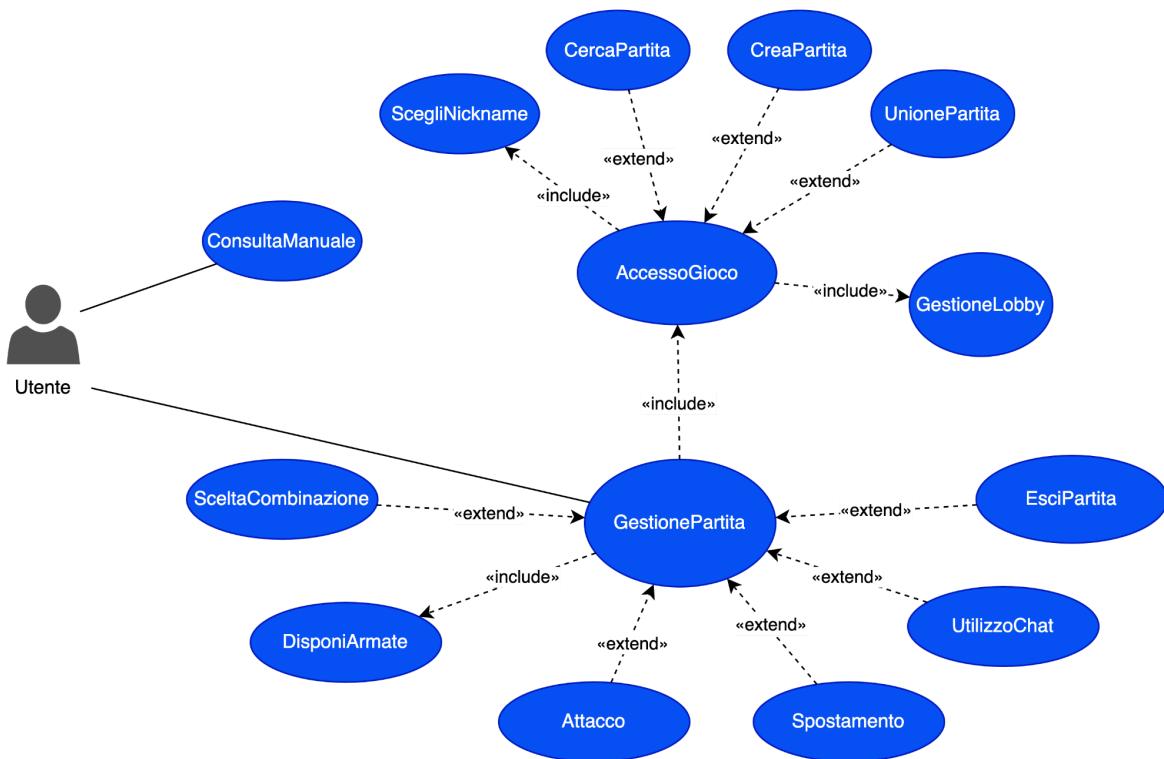
Analisi dei requisiti

Vocabolario

Voce	Definizione	Sinonimi
Partita	Concetto generale che racchiude la lobby e la sessione di gioco.	
Lobby	Stanza virtuale dove si raccolgono e gestiscono i giocatori.	Sala d'attesa
Sessione di gioco	Sfida tra i vari giocatori che competono per la vittoria.	
Utente	Persona che utilizza l'applicazione.	
Giocatore	È un utente che si trova in una lobby o in una sessione di gioco.	
Multigiocatore	Presenza di più giocatori nella partita.	
Tempo reale	Tutti i giocatori interessati partecipano alla partita nello stesso lasso di tempo senza subire ritardi notevoli nell'interazione.	
RISIKO	Gioco di società riguardante guerre tra diverse fazioni.	
Modalità ospite	Modalità di accesso in cui non è prevista l'autenticazione degli utenti	
Sistema	Software che effettua certe operazioni	Applicazione
Codice	Stringa identificativa di una partita	
Opzioni di una partita	Serie di proprietà che contraddistinguono una partita (mappa della partita, numero di giocatori in una partita, turni massimi giocabili e lingua della chat)	Opzioni di ricerca, Opzioni di creazione
Mappa	Insieme di regioni	

Voce	Definizione	Sinonimi
Regione	Insieme di territori	Continente (<i>mappa mondiale</i>), distretto/quartiere (<i>mappa cittadina</i>), etc...
Territorio	Unità atomica della mappa	
Creatore della partita	Giocatore che ha creato una partita	
Chat	Sezione dedicata alla messaggistica tra giocatori all'interno di una sessione di gioco	
Armata	Unità di attacco e difesa	Carro armato
Turno	Periodo di tempo in cui un singolo giocatore può partecipare attivamente	
Fase	Parte del turno in cui sono possibili opzioni differenti a seconda della fase considerata	
Manuale	Insieme illustrato delle regole del gioco con opportuni esempi	
Figura	Una immagine che rappresenta un cavallo, un fante, un cannone o un jolly	
Carta Territorio	Carta che comprende un territorio e una delle figure (tranne il jolly) per la scelta della combinazione.	
Carta Jolly	Carta speciale utilizzata nella scelta della combinazione, avente un jolly come figura.	
Carta Combinazione	È una carta territorio o una carta jolly.	

Diagramma dei casi d'uso



NOTA: Dopo varie considerazioni e dopo aver svolto il colloquio col prof, siamo giunti alla conclusione che Utente sia il solo attore. L'utente che effettua operazioni da giocatore verrà mostrato come “Utente(Giocatore)” e l’utente che svolge operazioni da creatore della partita verrà mostrato come “Utente(CreatoreDellaPartita)”.

Scenario: Consulta manuale

Titolo	ConsultaManuale
Descrizione	Viene visualizzato il manuale delle istruzioni di gioco di RISIKO
Attori	Utente
Relazioni	
Precondizioni	
Postcondizioni	Viene mostrato il manuale delle istruzioni
Scenario principale	<ol style="list-style-type: none"> 1. L'Utente richiede di mostrare il manuale di gioco 2. Il sistema recupera il manuale di gioco 3. Il sistema stampa a video il manuale di gioco 4. L'utente consulta il manuale di gioco
Scenari alternativi	
Requisiti non funzionali	
Punti aperti	

Scenario: Scegli nickname

Titolo	ScegliNickname
Descrizione	L'utente sceglie un nickname che sarà visibile durante il gioco
Attori	Utente
Relazioni	AccessoGioco
Precondizioni	
Postcondizioni	L'utente verrà associato ad un nickname, che verrà mostrato durante le partite
Scenario principale	<ol style="list-style-type: none"> 1. L'utente inserisce il nickname desiderato 2. Il sistema associa il nickname all'utente 3. Il sistema mostra la schermata delle modalità di accesso al gioco
Scenari alternativi	Scenario a: l'Utente inserisce nome inappropriate

	2. Il sistema mostra un messaggio errore e ripropone lo scenario dal punto 1
Requisiti non funzionali	
Punti aperti	

Scenario: Accesso gioco

Titolo	AccessoGioco
Descrizione	Vengono illustrate all'utente le tre modalità di accesso ad una partita supportate dall'applicazione e si occupa di far accedere l'utente ad una sessione di gioco
Attori	Utente
Relazioni	ScegliNickname, CercaPartita, CreaPartita, UnionePartita, GestionePartita, GestioneLobby
Precondizioni	L'utente è associato ad un nickname
Postcondizioni	L'utente è entrato in una sessione di gioco
Scenario principale	<p>L'utente sceglie una modalità di accesso a una partita</p> <ol style="list-style-type: none"> 1. ScegliNickname 2. Il Sistema presenta all'Utente le 3 modalità di accesso al gioco: accesso tramite ricerca di una partita, accesso tramite creazione di una partita e accesso tramite unione ad una partita 3. L'Utente sceglie una delle 3 modalità di accesso mostrate 4. Il Sistema porta l'Utente ad andare incontro a uno dei casi d'uso, in base alla scelta della modalità, tra: CercaPartita, CreaPartita e UnionePartita 5. GestioneLobby 6. Il Sistema inserisce gli Utenti(Giocatori) dentro la sessione di gioco della partita avviata
Scenari alternativi	
Requisiti non funzionali	
Punti aperti	

Scenario: Cerca Partita

Titolo	CercaPartita
Descrizione	L'utente cerca una partita in base alle opzioni di ricerca desiderate
Attori	Utente
Relazioni	AccessoGioco
Precondizioni	L'utente ha selezionato la modalità di accesso tramite ricerca di una partita
Postcondizioni	L'utente è riuscito ad accedere ad una lobby
Scenario principale	<ol style="list-style-type: none"> 1. L'Utente inserisce le opzioni di ricerca 2. Il sistema ricerca una partita utilizzando le opzioni di ricerca inserite e sposta l'utente in una lobby appropriata per l'attesa della fine della ricerca
Scenari alternativi	
Requisiti non funzionali	
Punti aperti	

Scenario: Crea Partita

Titolo	CreaPartita
Descrizione	L'utente crea una partita alla quale è associato un codice
Attori	Utente
Relazioni	AccessoGioco
Precondizioni	L'utente ha selezionato la modalità di accesso tramite creazione di una partita
Postcondizioni	L'utente è riuscito ad accedere ad una lobby
Scenario principale	<ol style="list-style-type: none"> 1. L'Utente inserisce le opzioni della partita 2. Il sistema crea una lobby, ovvero una partita non cominciata, che rispetta le opzioni inserite più un codice associato e affida la

	gestione all'Utente
Scenari alternativi	
Requisiti non funzionali	
Punti aperti	

Scenario: Unione Partita

Titolo	UnionePartita
Descrizione	L'utente si unisce ad una partita utilizzando un codice valido.
Attori	Utente
Relazioni	AccessoGioco
Precondizioni	L'utente ha selezionato la modalità di accesso tramite unione ad una partita
Postcondizioni	L'utente è riuscito ad accedere ad una lobby
Scenario principale	<ol style="list-style-type: none"> 1. L'Utente inserisce il codice di una partita 2. Il Sistema sposta l'Utente nella lobby associata al codice
Scenari alternativi	Scenario a: Codice di una partita non esistente o di una partita già cominciata 2. Il sistema mostra all'Utente un messaggio di errore appropriato e si riprende dal punto 1
Requisiti non funzionali	
Punti aperti	

Scenario: Gestione lobby

Titolo	GestioneLobby
Descrizione	Gli Utenti(Giocatori) vengono gestiti all'interno di una lobby nell'attesa dell'avviamento della partita
Attori	Utente
Relazioni	AccessoGioco
Precondizioni	L'utente ha utilizzato una delle tre modalità di accesso, quindi ha ricercato una partita oppure ha creato una partita oppure si è unito ad una partita
Postcondizioni	La partita è avviata.
Scenario principale	<ol style="list-style-type: none"> 1. L'Utente entra nella lobby 2. Viene assegnato un colore all'Utente(Giocatore) appena entrato 3. Eventualmente il sistema attende l'arrivo degli Utenti(Giocatori) necessari 4. Il Sistema o l'Utente(CreatoreDellaPartita) avviano la partita
Scenari alternativi	<p>Scenario a: L'Utente(CreatoreDellaPartita) espelle un Utente(Giocatore) dalla Lobby 4. L'Utente(CreatoreDellaPartita), invece di avviare la partita, espelle uno o più Utenti(Giocatori), si riprende dal punto 2</p> <p>Scenario b: L'Utente(CreatoreDellaPartita) scioglie la Lobby 4. L'Utente(CreatoreDellaPartita) scioglie la Lobby e tutti gli Utenti(Giocatori) vengono riportati alla scelta di modalità dell'accesso gioco</p> <p>Scenario c: Utente espulso precedentemente prova a rientrare nella stessa lobby 1. L'Utente visualizza un messaggio di errore che comunica il divieto di accesso alla lobby per il tempo di espulsione rimanente. Una volta che il tempo di espulsione si esaurisce si riprende dal punto 1 dello scenario principale</p> <p>Scenario d: Utente decide di abbandonare</p>

	<p>la Lobby.</p> <p>3. L'Utente(Giocatore) abbandona la Lobby e il Sistema riprende dal punto 3</p>
Requisiti non funzionali	
Punti aperti	

Scenario: Gestione partita

Titolo	GestionePartita
Descrizione	Insieme di azioni che svolgono gli Utenti(Giocatori) durante ogni loro turno della partita
Attori	Utente
Relazioni	AccessoGioco, SceltaCombinazione, DisponiArmate, Attacco, Spostamento, UtilizzoChat, EsciPartita
Precondizioni	L'utente è entrato in una sessione di gioco
Postcondizioni	La partita si è conclusa
Scenario principale	<ol style="list-style-type: none"> 1. AccessoGioco 2. Il Sistema assegna una carta obiettivo a ciascun giocatore. Il raggiungimento dell'obiettivo deve avvenire nei turni necessari oppure entro quelli specificati durante la ricerca o la creazione 3. Il Sistema distribuisce i territori ai vari Utenti(Giocatori) 4. Il Sistema distribuisce le armate iniziali a ciascun Utente(Giocatore) 5. Tutti gli Utenti(Giocatori) distribuiscono le armate ricevute nei territori posseduti 6. Inizia la partita che è composta da turni. In ciascun turno ci sono delle fasi in cui un Utente(Giocatore) può svolgere delle azioni. Ogni fase è associata ad un timer. Allo scadere di quest'ultimo il Sistema gestisce e fa terminare in automatico la fase stessa. Le Fasi sono: <ol style="list-style-type: none"> 6.1. Fase di scelta della combinazione: l'Utente(Giocatore) può scegliere se svolgere o meno SceltaCombinazione

	<p>6.2. Fase di disposizione delle armate: Il Sistema assegna le armate e l'Utente(Giocatore) esegue una o più volte DisponiArmata, finché non termina le armate da disporre che ha a disposizione</p> <p>6.3. Fase di attacco: l'Utente(Giocatore) esegue o meno una o più volte Attacco.</p> <p>6.4. Fase di spostamento: l'Utente(Giocatore) esegue o meno Spostamento. Alla fine di questa fase, se l'Utente(Giocatore) ha conquistato almeno un territorio durante la fase di attacco, il Sistema gli consegnerà una carta combinazione.</p> <p>7. L'Utente(Giocatore) può utilizzare eventualmente la chat per comunicare durante la partita</p> <p>8. La Partita termina perché un Utente(Giocatore) ha raggiunto il proprio Obiettivo oppure perché sono terminati i turni specificati della partita</p>
Scenari alternativi	<p>Scenario a: l'Utente(Giocatore) non riesce a distribuire le armate nel tempo limite</p> <p>6.2. Il Sistema distribuisce equamente le armate non distribuite dall'Utente(Giocatore), nei vari territori di quest'ultimo e si va al punto 7.3</p> <p>Scenario b: l'Utente(Giocatore) esce dalla partita</p> <p>(Può avvenire in qualunque punto dal 2 al 6). L'Utente(Giocatore) esce dalla partita e il Sistema distribuisce equamente le armate e i territori dell'Utente(Giocatore) che ha abbandonato in maniera appropriata. Si riprende dal punto in cui l'abbandono è avvenuto</p>
Requisiti non funzionali	
Punti aperti	

Scenario: Scelta combinazione

Titolo	SceltaCombinazione
Descrizione	L'utente sceglie 3 carte territorio, tra quelle che possiede, in modo da creare una combinazione tra quelle consentite
Attori	Utente
Relazioni	GestionePartita
Precondizioni	L'Utente(Giocatore) deve possedere 3 carte territorio valide per una combinazione
Postcondizioni	L'Utente(Giocatore) ha ottenuto le armate che gli spettano
Scenario principale	<p>1. Il Sistema mostra le combinazioni valide tra quelle che l'Utente(Giocatore) può scegliere</p> <p>2. L'Utente(Giocatore) sceglie le 3 carte per la combinazione e in base a quale combinazione gioca otterrà un certo numero di armate. Le combinazioni e le rispettive armate sono:</p> <ul style="list-style-type: none"> - 3 cannoni => 4 armate (*) - 3 fanti => 6 armate (*) - 3 cavalieri => 8 armate (*) - 1 fante, 1 cannone, 1 cavaliere => 10 armate (*) - 1 jolly e 2 carte uguali (2 cavalieri o 2 fanti o 2 cannoni) => 12 armate (*) <p>*Per ogni territorio che l'Utente(Giocatore) possiede tra quelli raffigurati nelle sue carte territorio vengono aggiunte altre 2 armate al premio</p>
Scenari alternativi	
Requisiti non funzionali	
Punti aperti	

Scenario: Disponi Armate

Titolo	DisponiArmate
Descrizione	L'Utente(Giocatore) è costretto a disporre un numero prestabilito di armate.
Attori	Utente
Relazioni	GestionePartita
Precondizioni	L'Utente(Giocatore) deve avere almeno un'armata da disporre
Postcondizioni	L'Utente(Giocatore) ha disposto le armate scelte per la disposizione
Scenario principale	<p>1. L'Utente(Giocatore) sceglie su quale territorio tra quelli che possiede vuole disporre delle armate</p> <p>2. L'Utente(Giocatore) sceglie quante armate disporre nel territorio scelto tra le armate disponibili</p> <p>N.B. L'Utente(Giocatore) ha un numero N di armate da disporre. Dove N è la somma di:</p> <ul style="list-style-type: none"> - Il numero di armate ricevute in base al numero di territori, ovvero numero di territori diviso 3, arrotondato per difetto. Ricevendo però sempre almeno un'armata. - Il numero di armate ricevute in base alle regioni che possiede. - Il numero di armate ricevute dalla combinazione
Scenari alternativi	
Requisiti non funzionali	
Punti aperti	

Scenario: Attacco

Titolo	Attacco
Descrizione	L'Utente(Giocatore) esegue un attacco da un territorio posseduto ad un territorio nemico confinante
Attori	Utente
Relazioni	GestionePartita

Precondizioni	Il territorio da cui l'Utente(Giocatore) attacca deve avere almeno 2 armate.
Postcondizioni	L'attacco è stato eseguito.
Scenario principale	<ol style="list-style-type: none"> 1. L'Utente(Giocatore) attaccante sceglie un territorio dal quale far partire un attacco 2. L'Utente(Giocatore) attaccante sceglie un territorio avversario confinante con il territorio scelto in precedenza 3. L'Utente(Giocatore) attaccante sceglie il numero di armate con cui vuole attaccare (da 1 a 3) lasciando almeno un'armata nel territorio attaccante 4. L'Utente(Giocatore) difensore sceglie il numero di armate con cui difendersi (da 1 a 3) in base a quelle che vuole utilizzare e in base a quelle che possiede 5. Entrambi gli Utenti(Giocatori) lanciano i dadi. 6. Durante l'esecuzione, il Sistema confronta i valori ottenuti dall'attaccante e dal difensore. I valori più alti vengono associati uno-a-uno, il più alto dell'attaccante con il più alto del difensore, il secondo con il secondo e così via. Per ogni confronto, il Sistema rimuove un'armata dal territorio del perdente (ovvero quello che ha ottenuto un valore dei dadi minore). In caso di parità, vince il difensore. Se il numero di armate dell'attacco e della difesa è diverso, il Sistema non considera nel confronto le armate con il valore di dadi minore dalla parte (attacco o difesa) che ha più armate fino a raggiungere un numero di armate equivalente. 7. Il Sistema mostra il risultato dello scontro. Lo scontro può avere vari esiti: <ul style="list-style-type: none"> a. Le armate rimanenti del difensore sono 0. Tutte le armate attaccanti residue dallo scontro sono spostate nel territorio del difensore. b. Le armate rimanenti del difensore sono diverse da 0, quindi il territorio non è stato

	conquistato.
Scenari alternativi	
Requisiti non funzionali	
Punti aperti	

Scenario: Spostamento

Titolo	Spostamento
Descrizione	L'Utente(Giocatore) può spostare un certo numero di armate da un suo territorio ad un altro suo territorio, confinante con il primo
Attori	Utente
Relazioni	GestionePartita
Precondizioni	L'Utente(Giocatore) deve avere più di una armata nel territorio dal quale vuole far partire lo spostamento
Postcondizioni	L'Utente(Giocatore) ha effettuato lo spostamento
Scenario principale	<ol style="list-style-type: none"> 1. L'Utente(Giocatore) seleziona un territorio posseduto dal quale far partire lo spostamento 2. L'Utente(Giocatore) seleziona un territorio posseduto confinante al primo, come destinazione dello spostamento 3. L'Utente(Giocatore) sceglie il numero di armate che intende spostare, lasciando almeno un armata dal territorio da cui parte lo spostamento 4. Il Sistema esegue lo spostamento
Scenari alternativi	
Requisiti non funzionali	
Punti aperti	

Scenario: Utilizzo chat

Titolo	UtilizzoChat
Descrizione	Viene data la possibilità di poter comunicare all'interno della partita tra i vari Utenti(Giocatori) partecipanti
Attori	Utente
Relazioni	GestionePartita
Precondizioni	L'Utente(Giocatore) è all'interno di una Sessione di Gioco
Postcondizioni	L'Utente(Giocatore) è riuscito a comunicare con gli altri Utenti(Giocatori) all'interno della partita utilizzando la chat apposita
Scenario principale	<ol style="list-style-type: none"> 1. L'Utente(Giocatore) scrive un messaggio nella chat 2. Il Sistema mostra il messaggio nella chat di tutti gli Utenti(Giocatori)
Scenari alternativi	
Requisiti non funzionali	
Punti aperti	

Scenario: Esci partita

Titolo	EsciPartita
Descrizione	L'Utente(Giocatore) esce dalla partita
Attori	Utente
Relazioni	GestionePartita
Precondizioni	L'Utente(Giocatore) è all'interno di una Sessione di Gioco
Postcondizioni	L'Utente(Giocatore) è uscito dalla Partita
Scenario principale	<ol style="list-style-type: none"> 1. L'Utente(Giocatore) esce dalla partita 2. Il Sistema mostra agli altri Utenti(Giocatori) l'abbandono dell'Utente(Giocatore) in questione
Scenari alternativi	
Requisiti non funzionali	

Titolo	EsciPartita
Punti aperti	

Analisi del Rischio

Tabella Valutazione dei Beni

Bene	Valore	Esposizione
Partita	Basso. Contiene la gestione della lobby e tutte le possibili azioni che possono compiere gli Utenti(Giocatori)	Bassa. La modifica di una partita intralicia il suo corretto svolgimento solo per i suoi Utenti(Giocatori)
Mappe e file di configurazione	Medio. Dati informativi per la generazione delle partite	Media. La perdita o danneggiamento delle mappe impedisce di giocare a tutti gli utenti
Informazioni utenti	Basso. Si tratta solo di un nickname variabile	Bassa. Un nickname compromesso potrebbe creare confusione nella partita

Tabella Minacce/Controlli

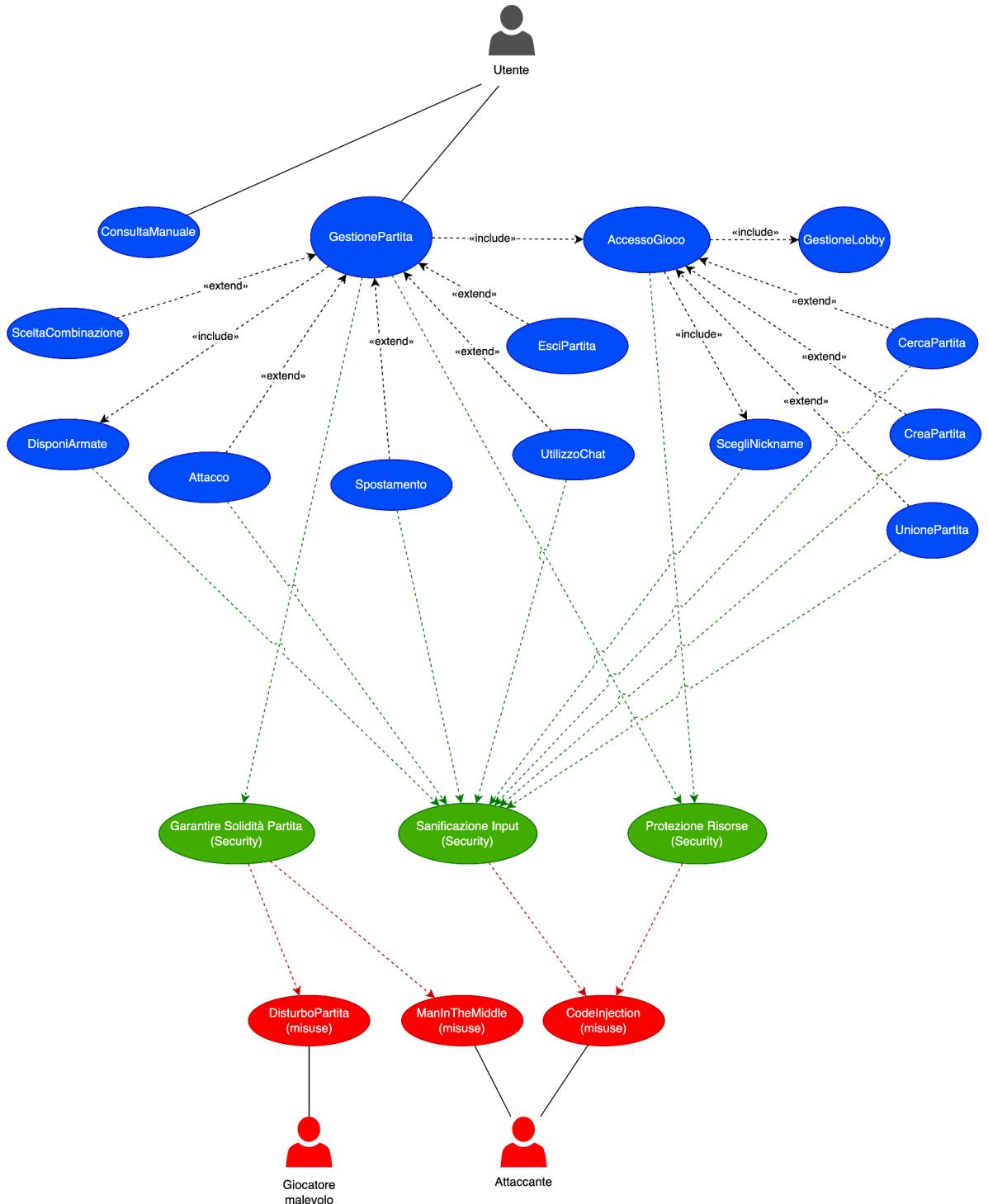
Minaccia	Probabilità	Controllo	Fattibilità
Modifica delle mappe e dei file di configurazione	Bassa. Modifiche inconsistenti delle mappe da parte di attaccanti compromettono la giocabilità	Progettazione adeguata, controllo dell'input, limitazione delle risorse disponibili e utilizzare protocolli sicuri per la comunicazione	Basso costo. Usiamo degli standard già definiti e supportati per cifrare le comunicazioni
Disturbo della partita	Bassa. Le modifiche improprie di stato della partita manomettono il corretto svolgimento della stessa	Progettazione del sistema adeguato, che verifica possibili manomissioni dello stato della partita	Basso costo Implementativo
		Log di gestione delle partite	Basso costo Implementativo

DoS	Bassa	Progettazione adeguata, controllo e limitazione degli accessi	Basso costo. Impossibile prevenire un DoS
-----	-------	---	--

Analisi Tecnologica della Sicurezza

Tecnologia	Vulnerabilità
Accesso con nickname	<ul style="list-style-type: none"> • Non è univoco • Non è controllabile
Architettura Client/Server	<ul style="list-style-type: none"> • DoS • Man in the Middle
Cifratura comunicazioni	<ul style="list-style-type: none"> • Utilizzo di una cifratura ibrida, simmetrica e asimmetrica, con le loro relative vulnerabilità: • Cifratura Simmetrica: <ul style="list-style-type: none"> ◦ Tempo di vita della chiave ◦ Più informazioni vengono cifrate con la stessa chiave, più materiale è offerto per l'analisi del testo a un attaccante ◦ Memorizzazione della chiave ◦ Lunghezza della chiave • Cifratura Asimmetrica: <ul style="list-style-type: none"> ◦ Lunghezza della chiave ◦ Memorizzazione della chiave privata. L'utilizzo di cifrature standard non recenti potrebbero causare vulnerabilità ulteriori.

Security Use Case & Misuse Case



Security Use Case & Misuse Case Scenari

Titolo	ProtezioneRisorse	
Descrizione	Le risorse del sistema devono essere immodificabili dall'esterno	
Misuse case	CodeInjection	
Relazioni		
Precondizioni	L'attaccante può iniettare codice malevolo e modificare il file system dell'applicativo	
Postcondizioni	Il sistema tiene traccia del tentativo di modifica delle risorse eseguendo un eventuale backup di tali risorse	
Scenario principale	Sistema	Attaccante
		L'attaccante lancia un attacco inserendo codice malevolo
	Il sistema rifiuta i messaggi mal formattati e ne tiene traccia in un log	
Scenario di attacco avvenuto con successo	Sistema	Attaccante
		L'attacco è riuscito
	Il sistema continua a tenere traccia delle operazioni illecite eseguite	
		L'attaccante corrompe le risorse essenziali per l'applicativo
	Il sistema tiene traccia di qualsiasi modifica apportata alle risorse	
	Periodicamente, il sistema può eseguire un backup delle risorse statiche, negando gli effetti duraturi dell'attacco	

Titolo	GarantireSoliditàPartita	
Descrizione	Bisogna proteggere l'esperienza di gioco da malintenzionati	
Misuse case	DisturboPartita, ManInTheMiddle	
Relazioni		
Precondizioni	L'attaccante è in grado di modificare il naturale flusso di gioco e le proprietà di una partita	
Postcondizioni	Il sistema previene disturbi della partita garantendo la solidità dello stato della stessa ed espelle gli Utenti(Giocatori) malevoli	
Scenario principale	Sistema	Attaccante
		L'Utente(Giocatore) prova ad intervenire sulle sessioni di gioco sue e/o degli altri per truccare il gioco
	Il sistema controlla che gli interventi nella partita vengano compiuti solo da chi dovrebbe averne diritto	
	Il sistema controlla l'integrità dello stato della partita e proibisce azioni che lo possono manomettere	
Scenario di attacco avvenuto con successo	Sistema	Attaccante
		L'attaccante prova ad intervenire nella sessione sua o di un altro Utente(Giocatore) quando e come non dovrebbe

	Il sistema gli consente l'intervento tramite una falla al suo interno	
	L'attaccante ha accesso a operazioni che non gli sarebbero consentite e inizia ad usarle a suo favore	
	Il sistema conserva un log di partita.	

Titolo	SanificazioneInput	
Descrizione	Gli input dell'utente devono essere sanificati	
Misuse case	CodeInjection	
Relazioni		
Precondizioni	L'attaccante ha i mezzi per poter trovare i vari input utente non protetti da controlli e non sanificati	
Postcondizioni	Il sistema previene tentativi di code injection sanificando l'input prima dell'utilizzo	
Scenario principale	Sistema	Attaccante
		Dopo aver scoperto punti di input utente, prova a iniettare codice malevolo per manomettere il sistema o parte di esso
	Controlla l'input utente e impedisce che del codice malevolo possa essere iniettato utilizzando i campi di input che il sistema dispone	
Scenario di attacco avvenuto con successo	Sistema	Attaccante
		L'attacco di code injection avviene

		con successo
	Il sistema non si accorge dell'input malevolo e consente le operazioni che l'attaccante intendeva svolgere	
	Naviga nel sistema e svolge operazioni che non gli sarebbero concesse	
	Il sistema scrive nel log tutte le operazioni eseguite dall'utente	

Requisiti di Protezione dei Dati

Dall'analisi del rischio si possono evincere i seguenti ulteriori requisiti:

1. Creazione di un log per tracciare
 - a. tutte le azioni che avvengono durante una partita
 - b. tutte le azioni di accesso a risorse
2. I dati memorizzati e scambiati nel sistema devono essere protetti
3. Gli input dell'utente vanno sanificati

Tabella dei requisiti

Nota: requisiti aggiunti ai requisiti già esistenti

ID	Requisito	Tipo
R15F	Creazione di un log per tracciare le operazioni	Funzionale
R16F	Protezione dei dati memorizzati e scambiati nel sistema	Funzionale
R17F	Gli input dell'utente devono essere sanificati	Funzionale

Si noti come il log creato in R15F non venga analizzato utilizzando funzionalità dal sistema, poiché questa operazione verrà svolta da un gestore dei server.

Si noti inoltre che i dati memorizzati, citati in R16F, siano solo i dati di una partita solo nell'arco della sua durata.

Analisi

Analisi Documento dei Requisiti: Analisi delle Funzionalità

Tabella Funzionalità

Funzionalità	Tipo	Grado Complessità	Requisiti collegati
ScegliNickname	Interazione tra giocatori, memorizzazione dati	semplice	R2F, R17F
ConsultaManuale	Gestione dati	semplice	R14F
AccessoGioco	Gestione dati	complessa	R4F, R5F, R6F, R7F, R8F, R9F, R1tF
GestionePartita	Interazione tra giocatori, gestione dati	complessa	R1F, R3F, R10F, R12F, R13F, R16F, R17F
ScritturaLog	Memorizzazione dati	semplice	R15F

ScegliNickname: Tabella Informazioni/Flusso

Informazione	Tipo	Livello protezione / privacy	Input / Output	Vincoli
Nickname	semplice	bassa	Input e Output	Non più di 16 caratteri

ConsultaManuale: Tabella Informazioni/Flusso

Informazione	Tipo	Livello protezione / privacy	Input / Output	Vincoli
Testo manuale <u>Composto da:</u> Testo	composto	bassa	Output	
Immagini	semplice	bassa	Output	
	semplice	bassa	Output	

AccessoGioco: Tabella Informazioni/Flusso

Informazione	Tipo	Livello protezione / privacy	Input / Output	Vincoli
Opzioni partita	composto	bassa	Input	
<u>Composto da:</u> Nome mappa	semplice	bassa	Input	Enumeratore: MONDO, EUROPA, BOLOGNA
Numero giocatori	semplice	bassa	Input	Da 3 a 6
Turni massimi	semplice	bassa	Input	Minimo 10
Preferenza lingua	semplice	bassa	Input	Enumeratore: IT, EN
Codice partita	semplice	media	Input e Output	Lunga 6 caratteri alfanumerici
Nome creatore della partita	semplice	bassa	Output	Non più di 16 caratteri
Lista giocatori nella lobby	composto	bassa	Output	Per ognuno, non più di 16 caratteri

GestionePartita: Tabella Informazioni/Flusso

Informazione	Tipo	Livello protezione / privacy	Input / Output	Vincoli
Immagine Mappa	semplice	bassa	Output	Solo una per tutta la partita
Giocatore	composto	bassa	Output	Minimo 3 e massimo 6
<u>Composto da:</u> Nome	semplice	bassa	Output	Non più di 16 caratteri
Colore	semplice	bassa	Output	Enumeratore: GIALLO, ROSSO, VERDE, BLU,

				VIOLA, NERO
Carta obiettivo (posseduta)	semplice	media	Output	Massimo 1 carta per giocatore
Carta combinazione (posseduta) <u>Composta da:</u> Nome figura	composta	media	Output	
Nome territorio	semplice	media	Output	È vuoto se la figura è un jolly
Combinazione <u>Composto da:</u> 3 carte combinazione possedute Numero armate ottenibili	composta composta semplice	media media media	Input e Output Input e Output Input e Output	
Regione <u>Composto da:</u> Nome Numero di armate bonus Nome possessore	composto semplice semplice semplice	basso bassa bassa bassa	Output Output Output Output	Vuoto se non c'è possessore
Territorio <u>Composto da:</u> Nome Numero di armate Nome possessore	composto semplice semplice semplice	basso bassa bassa bassa	Output Output Output Output	
Disposizione <u>Composta da:</u> Nome territorio	composta semplice	bassa bassa	Input e Output Input e Output	

Numero armate	semplice	bassa	Input e Output	
Attacco <u>Composto da:</u> Nome territorio attaccante	composta semplice	bassa bassa	Input e Output Input e Output	
Numero armate attaccanti	semplice	bassa	Input e Output	Compreso tra 1 e 3
Nome territorio difensore	semplice	bassa	Input e Output	
Numero armate difensore	semplice	bassa	Input e Output	Compreso tra 1 e 3
Spostamento <u>Composto da:</u> Nome territorio partenza	composta semplice	bassa bassa	Input e Output Input e Output	
Nome territorio arrivo	semplice	bassa	Input e Output	
Messaggi <u>Composto da:</u> Nome giocatore	semplice semplice	bassa bassa	Input e Output Output	
Testo	semplice	bassa	Input e Output	

ScritturaLog: Tabella Informazioni/Flusso

Informazione	Tipo	Livello protezione / privacy	Input / Output	Vincoli
Data	semplice	bassa	Input	
Ora	semplice	bassa	Input	
Operazione eseguita	semplice	bassa	Input	
Messaggio <u>Composto da:</u> Testo	composto semplice	bassa bassa	Input Input	

Nome giocatore	semplice	bassa	Input	
----------------	----------	-------	-------	--

Analisi Documento dei Requisiti: Analisi dei Vincoli

Tabella Vincoli

Requisito	Categorie	Impatto	Funzionalità
Velocità delle comunicazioni	Tempo di risposta	Obbliga ad avere limitate scelte progettuali, ma permette la giocabilità in tempo reale e una chat	GestionePartita
Protezione delle comunicazioni	Sicurezza	Peggiorano tempo di risposta, migliorano la protezione dei dati	AccessoGioco, GestionePartita

Analisi Documento dei Requisiti: Analisi Interazioni

Tabella Maschere

Maschera	Informazioni	Funzionalità
View Consulta Manuale	Illustra tutte le informazioni riguardanti il gioco	ConsultaManuale
Home Applicazione	Espone le possibilità di scelta del nickname e di accesso al gioco (solo se si è già scelto il nickname)	AccessoGioco
View Imposta Nickname	Offre all'utente la possibilità di inserire o cambiare un nickname	ScegliNickname
View Accesso	Espone all'utente le tre possibili modalità di accesso al gioco	AccessoGioco
View Cerca	Chiede all'utente di inserire le opzioni di ricerca di una partita	CercaPartita
View Crea	Chiede all'utente di inserire	CreaPartita

	le opzioni della partita da creare	
View Unione	Chiede all'utente di inserire il codice di una partita già creata ma non cominciata	UnionePartita
View Gestione Lobby	Illustra al creatore della partita quali giocatori che si sono uniti attraverso il codice e le possibili azioni effettuabili per gestire la sala d'attesa	GestioneLobby (da CreaPartita)
View Lobby	Illustra all'utente gli altri utenti che si sono uniti alla partita	GestioneLobby (da UnionePartita e CercaPartita)
View Gioco	Illustra al giocatore la mappa della partita, tutte le armate, l'elenco dei giocatori e il loro colore. Ogni giocatore può rivedere il suo obiettivo e le sue carte territorio. Sempre illustrata la chat a lato della mappa. Viene illustrata, per ogni giocatore il numero di territori posseduti, il numero di armate in ogni territorio e il numero di carte territorio possedute.	GestionePartita
View Combinazione	Illustra al giocatore le possibili combinazioni giocabili e il numero di armate che ne deriverebbe	SceltaCombinazione
View Disposizione	Illustra al giocatore il numero di armate schierabili rimaste e i possibili territori sui quali possono essere schierate	DisponiArmate
View Attacco	Evidenzia al giocatore tutti i territori attaccabili da un suo territorio selezionato (solo se l'attacco è eseguibile) In seguito alla scelta, viene evidenziato l'attaccante, il territorio da cui parte l'attacco, il difensore e il territorio attaccato.	Attacco

	<p>Viene inserito il numero di armate utilizzabili dall'attaccante.</p> <p>Il difensore ha un ruolo attivo, scegliendo anche lui il numero di armate con il quale può difendersi.</p> <p>Viene illustrato il lancio di dadi e il risultato dello scontro.</p> <p>Per gli altri giocatori all'interno della partita viene evidenziato il giocatore attaccante, il territorio da cui origina l'attacco e il numero di armate impiegate. Viene illustrato il lancio dei dadi e il risultato dello scontro</p>	
View Spostamento	Evidenzia al giocatore tutti i suoi territori nei quali è possibile eseguire uno spostamento tattico dopo aver scelto un territorio di origine di tale spostamento	Spostamento
View Esci	Viene illustrato un messaggio di uscita della partita	EsciPartita

Tabella Sistemi Esterni

Non sono presenti sistemi esterni.

Analisi Ruoli e Responsabilità

Tabella Ruoli

Ruolo	Responsabilità	Maschere	Riservatezza	Numerosità
Utente	Scelta del proprio nickname e accesso a una partita	View Manuale, Home Applicazione, View Nickname, View Accesso, View Cerca, View Unione, View Crea	Livello basso	Non c'è un numero massimo logico di utenti
Utente (Creatore della partita)	Gestione della lobby che crea.	View Manuale, View Gestione Lobby	Livello basso	Al massimo uno per lobby
Utente (Giocatore)	Deve seguire le regole del gioco e giocare correttamente	View Manuale, View Lobby, View Gioco, View Combinazione, View Disposizione, View Attacco, View Spostamento, View Esci	Livello medio	Da 1 a 6 giocatori massimo per ogni Lobby. Da 3 a 6 giocatori massimo per ogni partita cominciata.

Utente: Tabella Ruolo-Informazione

Informazione	Tipo Accesso
Nickname	Lettura/Scrittura
Testo manuale	Lettura
Opzioni della partita	Scrittura
Codice partita	Scrittura

Utente(Creatore della partita): Tabella Ruolo-Informazione

Informazione	Tipo Accesso
Nickname	Lettura
Testo manuale	Lettura

Opzioni della partita	Lettura
Codice partita	Lettura
Lista giocatori nella lobby	Lettura/Scrittura

Utente(Giocatore): Tabella Ruolo-Informazione

Informazione	Tipo Accesso
Nickname	Lettura
Testo manuale	Lettura
Opzioni della partita	Lettura
Codice partita	Lettura
Giocatori	Lettura
Immagine Mappa	Lettura
Carta obiettivo posseduta	Lettura
Carte combinazione possedute	Lettura/Scrittura
Combinazioni valide	Lettura
Territori	Lettura
Disposizioni	Lettura/Scrittura
Attacchi	Lettura/Scrittura
Spostamenti	Lettura/Scrittura
Messaggi	Lettura/Scrittura

Scomposizione del Problema

Tabella Scomposizione Funzionalità

Funzionalità	Scomposizione
AccessoGioco	CercaPartita, CreaPartita, UnionePartita, GestioneLobby
GestionePartita	SceltaCombinazione,

	DisponiArmate, Attacco, Spostamento, UtilizzoChat, EsciPartita
--	--

Tabella Sotto - Funzionalità

Sotto-Funzionalità	Sotto-Funzionalità	Legame	Informazioni
CercaPartita	GestioneLobby	GestioneLobby dipende da CercaPartita, per gli utenti che si uniscono alla partita tramite CercaPartita	Nickname
CreaPartita	GestioneLobby	GestioneLobby dipende da CreaPartita, per gli utenti con ruolo “Creatore della partita”	Nickname
UnionePartita	GestioneLobby	GestioneLobby dipende da UnionePartita, per gli utenti che si uniscono alla partita tramite UnionePartita	Nickname
SceltaCombinazione	DisponiArmate	Prima di disporre le armate, bisogna decidere, se si ha la possibilità e la volontà, di utilizzare una combinazione di carte combinazione possedute per ricevere delle armate in cambio	Numero di armate ottenute con combinazione
DisponiArmate	Attacco	Non si può attaccare se prima non sono state disposte le armate	Mappa aggiornata dopo le disposizioni
Attacco	Spostamento	Lo spostamento avviene sempre dopo la fase d'attacco, se si ha	Mappa aggiornata dopo gli attacchi

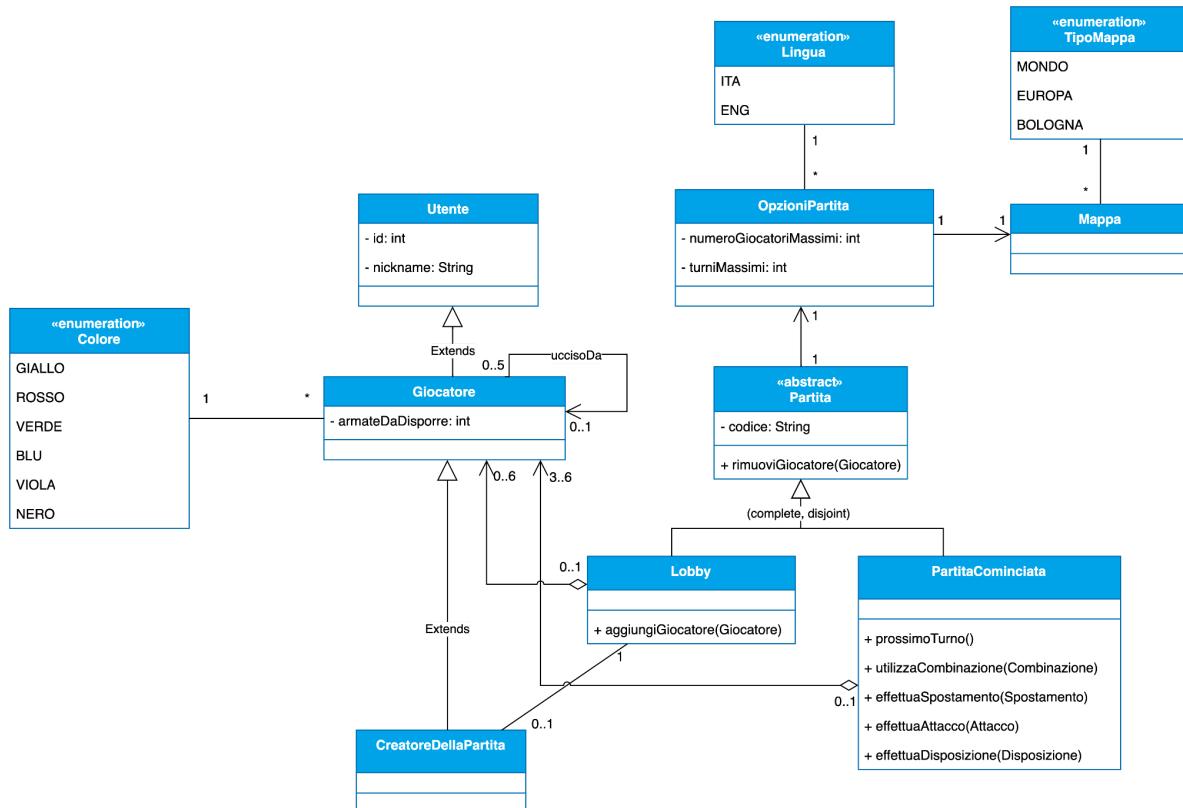
		deciso di svolgerne qualcuno	
--	--	------------------------------	--

Creazione modello del dominio

Il seguente diagramma delle classi rappresenta la parte del modello del dominio relativa alla partita e alle sue componenti.

Per semplificare la lettura abbiamo suddiviso il dominio della Partita in 3 parti dove le relazioni e gli attributi, una volta definiti, non verranno riportati nei ritagli successivi:

1) Definizione generica della partita



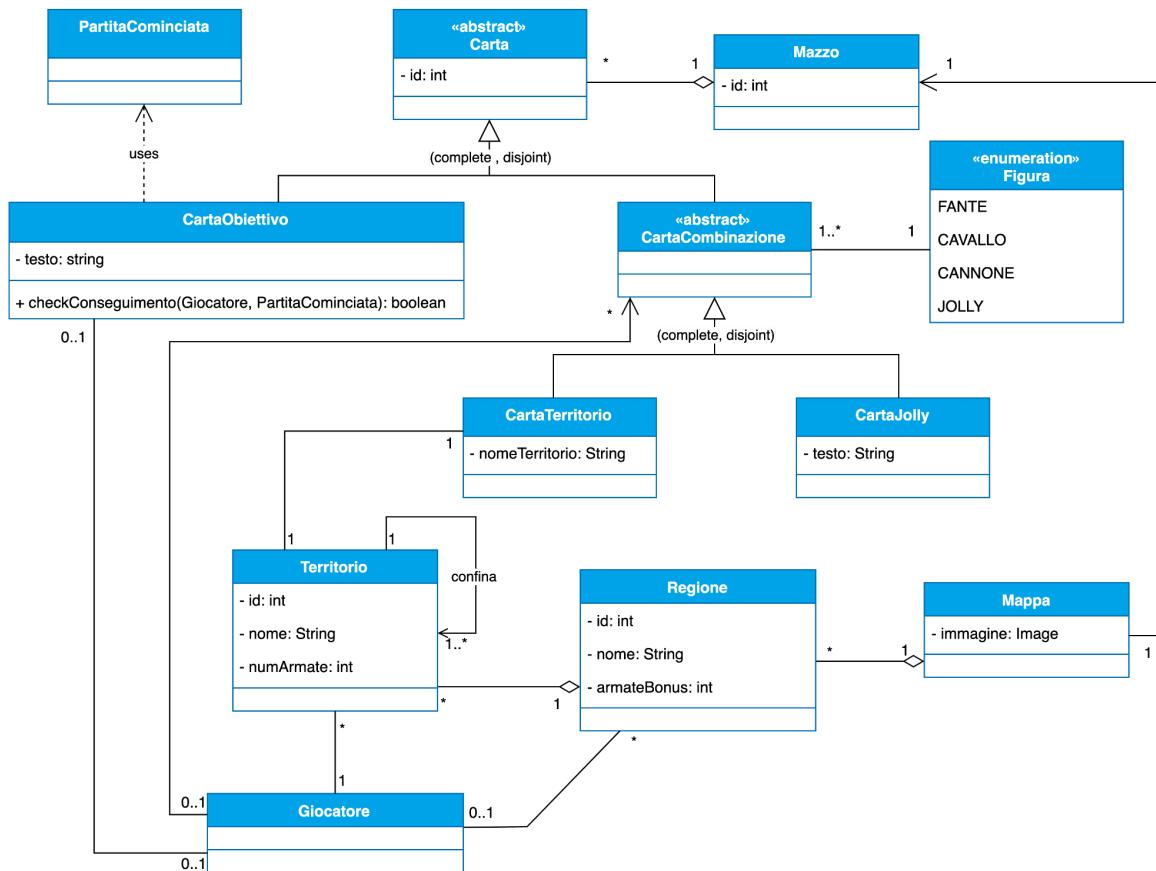
(1) N.B: Nel contesto del sistema un colore può essere associato a più Giocatori, ma mai a più di un Giocatore nella stessa partita.

(2) N.B: La cardinalità scelta per Giocatore a Giocatore col verbo “uccisoDa” è stata scelta dopo aver ragionato nel caso opposto, ovvero che un giocatore può uccidere da 0 a 5 Giocatori e può essere ucciso da al massimo un Giocatore.

(3) N.B: “armateDaDisporre” si azzera per forza entro la fine una fase di disposizione del giocatore in questione.

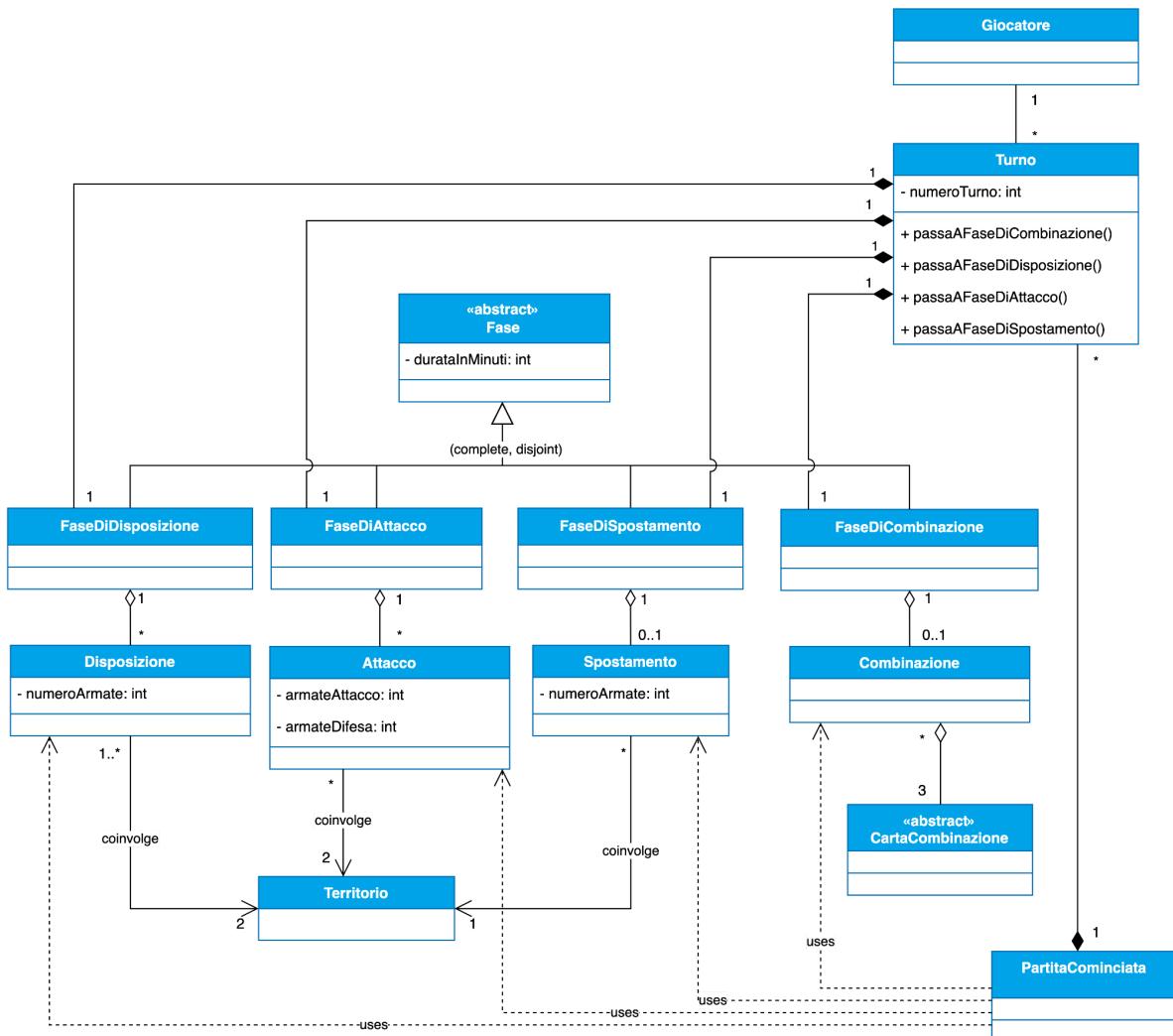
(4) N.B: Quando un Giocatore è dentro a una Lobby non può essere associato a nessuna PartitaCominciata e viceversa.

2) Gli elementi presenti nella partita

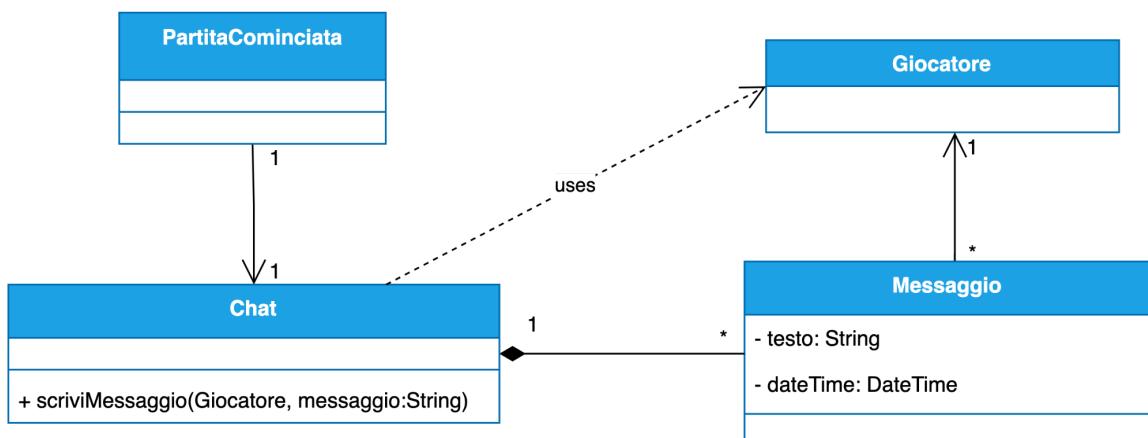


N.B. Consideriamo una Regione conquistata da un Giocatore solo se questo possiede tutti i territori che compongono tale Regione.

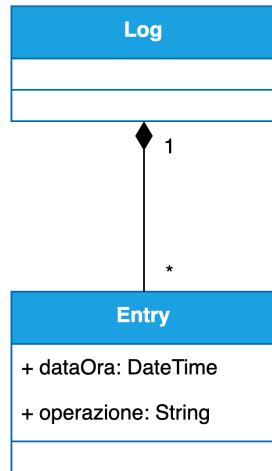
3) La strutturazione dei Turni e delle Fasi



4) La chat



Il seguente diagramma delle classi rappresenta la parte del modello del dominio relativa a Log.



Architetture Logica: Struttura

Diagramma dei package

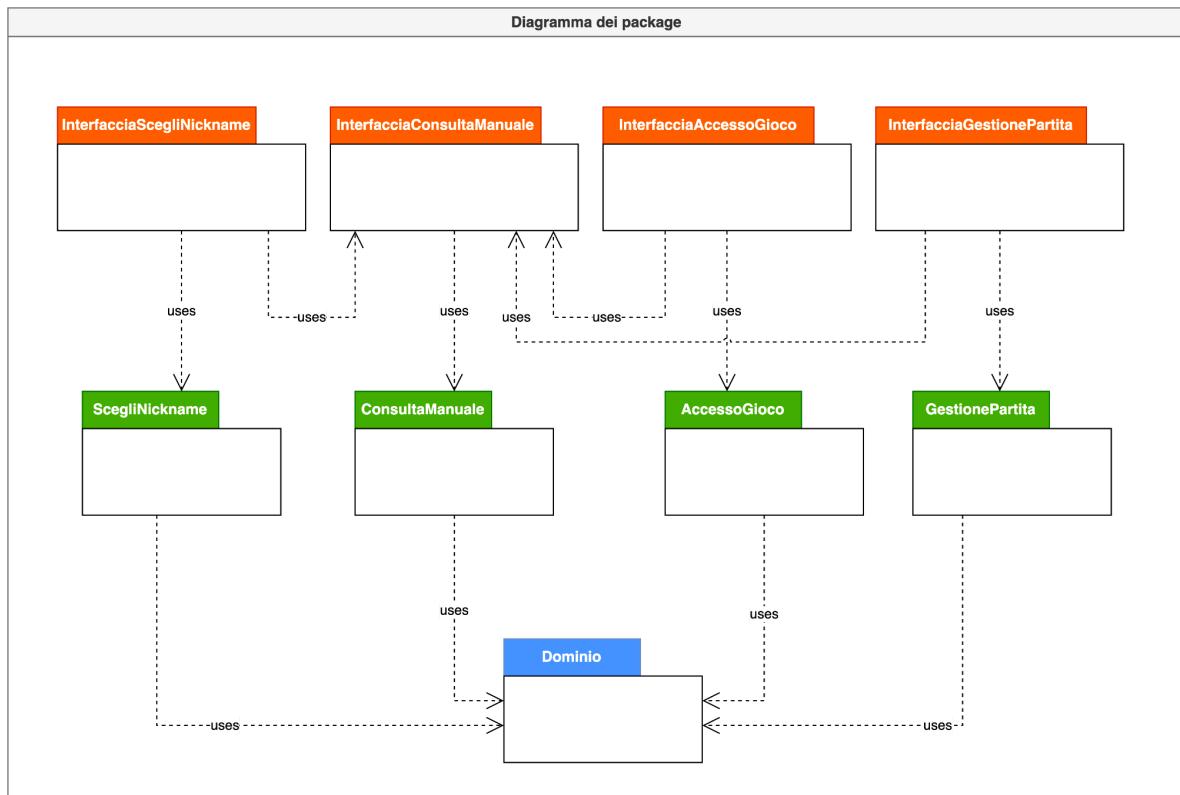


Diagramma delle classi: Dominio

Il Diagramma delle classi del Dominio è già stato definito nella fase precedente, quindi non viene riportato.

Diagramma delle classi: ScegliNickname & InterfacciaScegliNickname

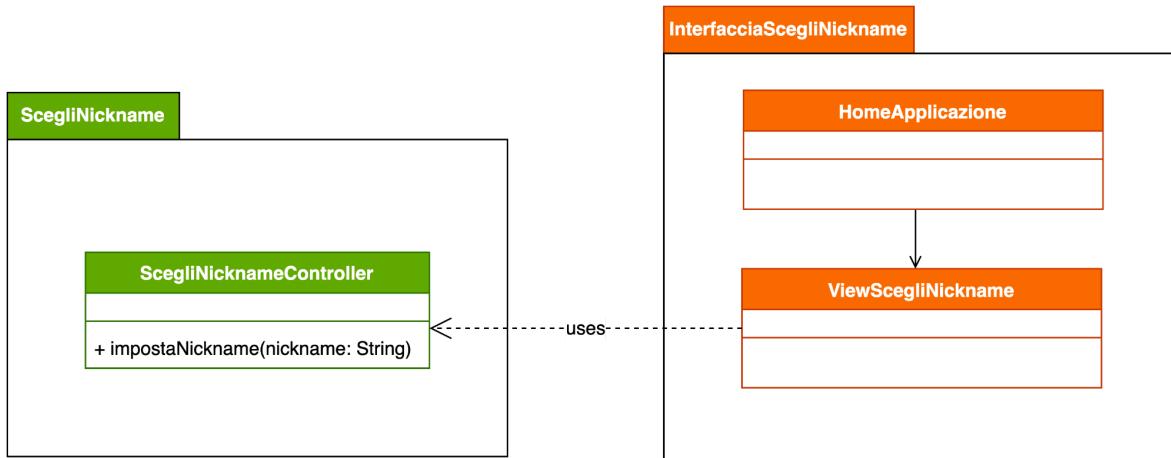


Diagramma delle classi: ConsultaManuale & InterfacciaConsultaManuale

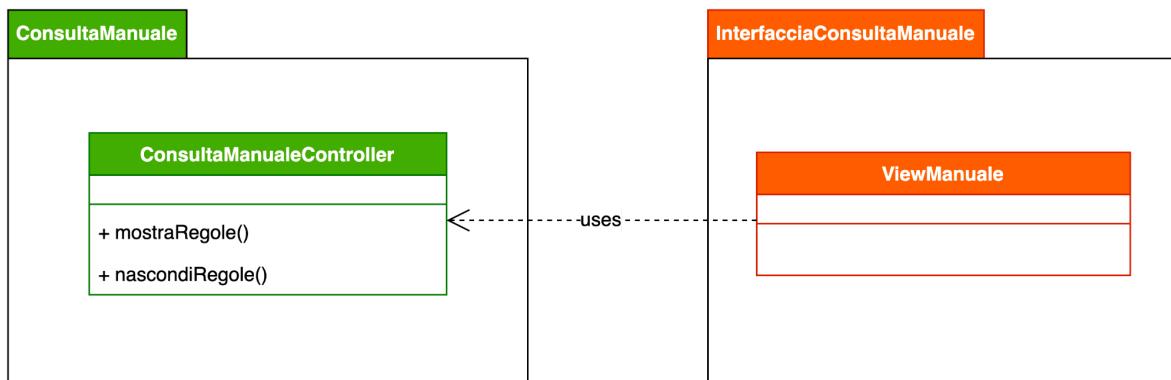


Diagramma delle classi: AccessoGioco & InterfacciaAccessoGioco

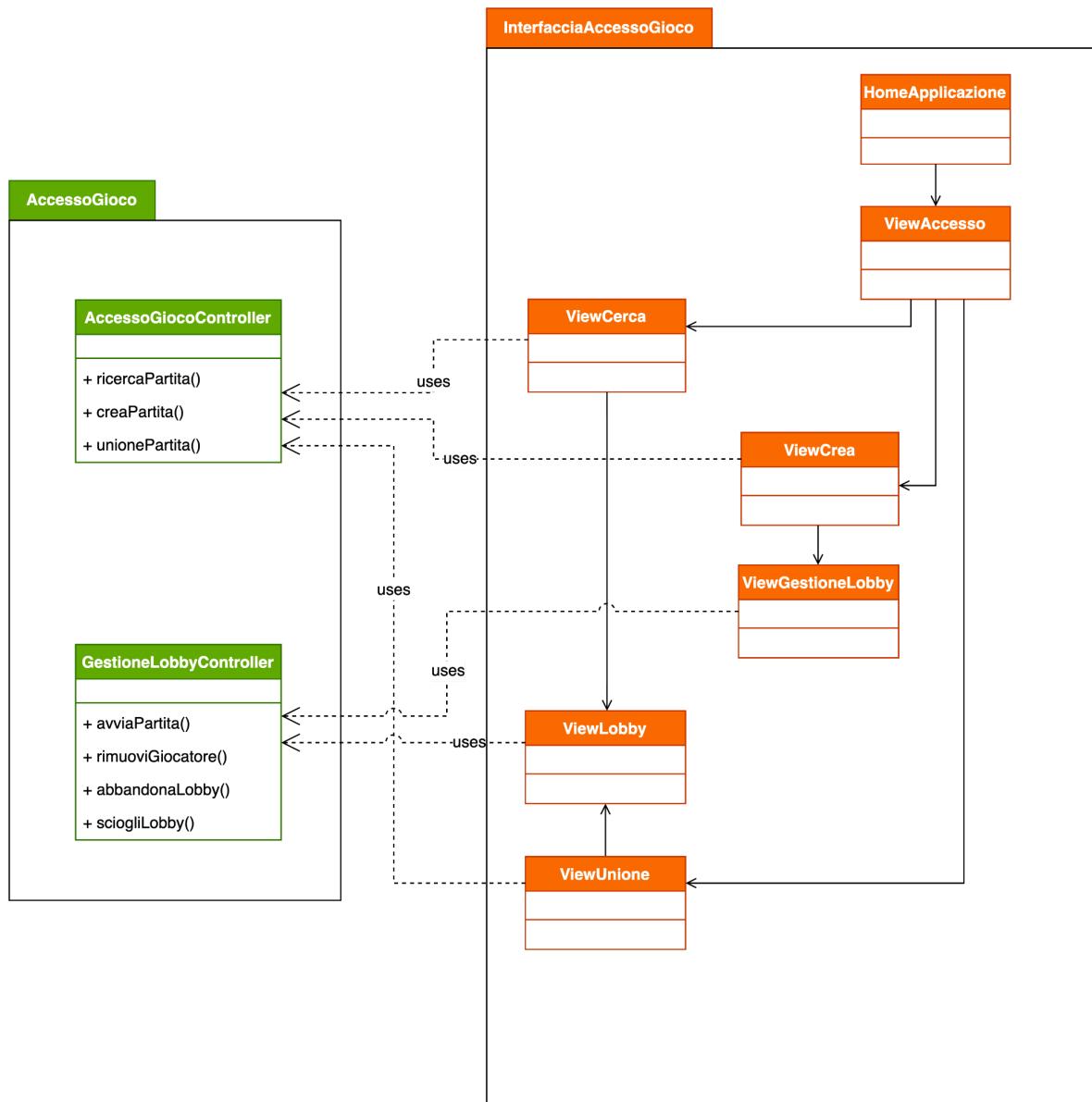
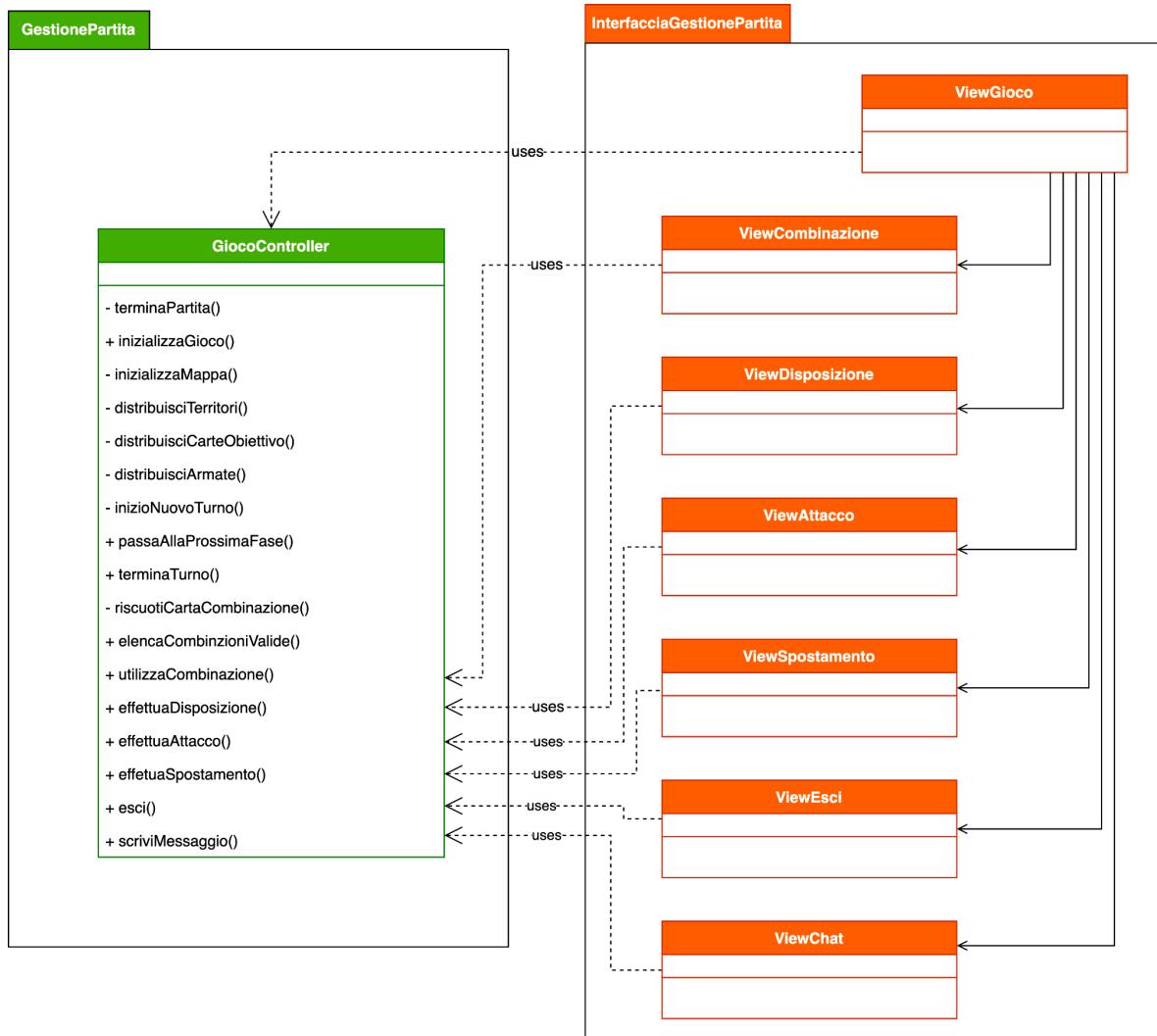
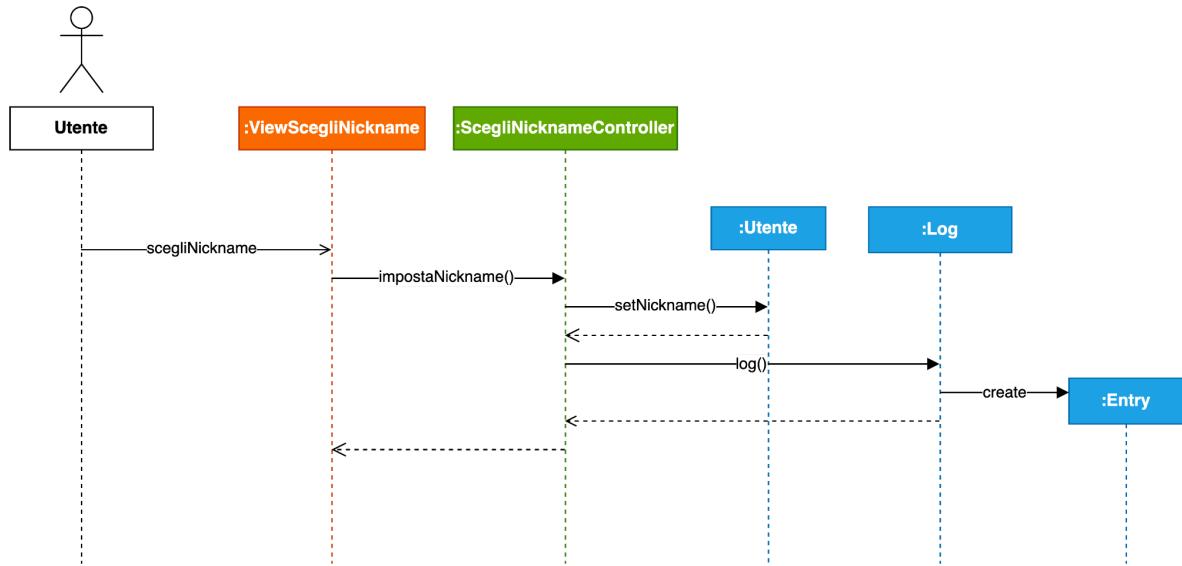


Diagramma delle classi: GestionePartita & InterfacciaGestionePartita



Architetture Logica: Interazione

Diagramma di sequenza: Impostazione del Nickname



NOTA: "Utente" non viene creato in nessuna sequenza perché è compito del sistema creare uno quando un utente si collega al client dell'applicazione.

Diagramma di sequenza: Visualizza Manuale

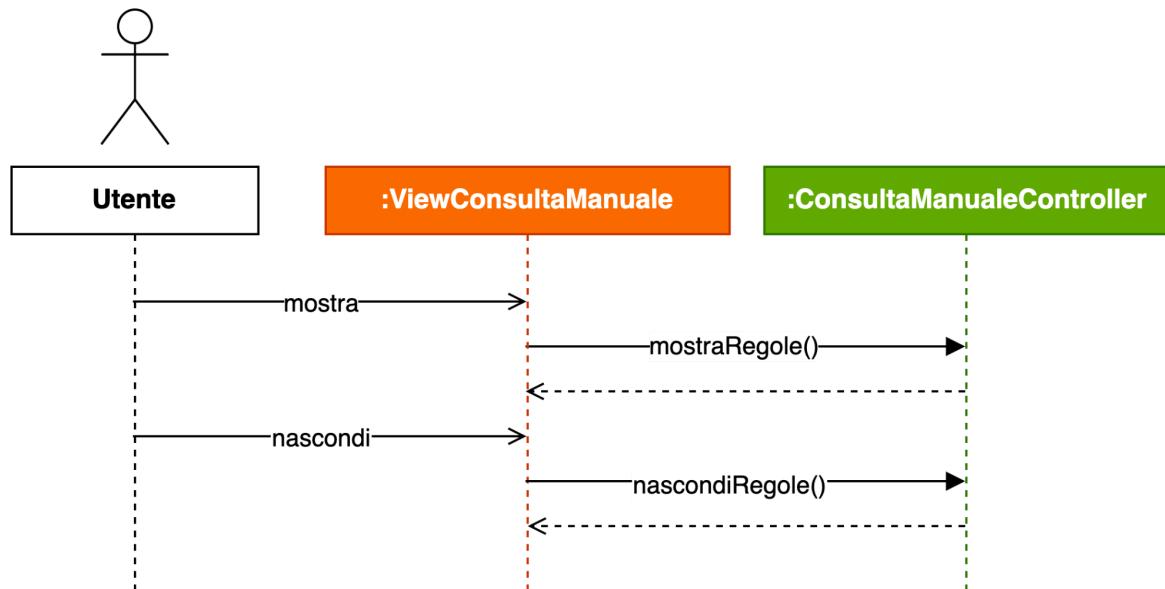
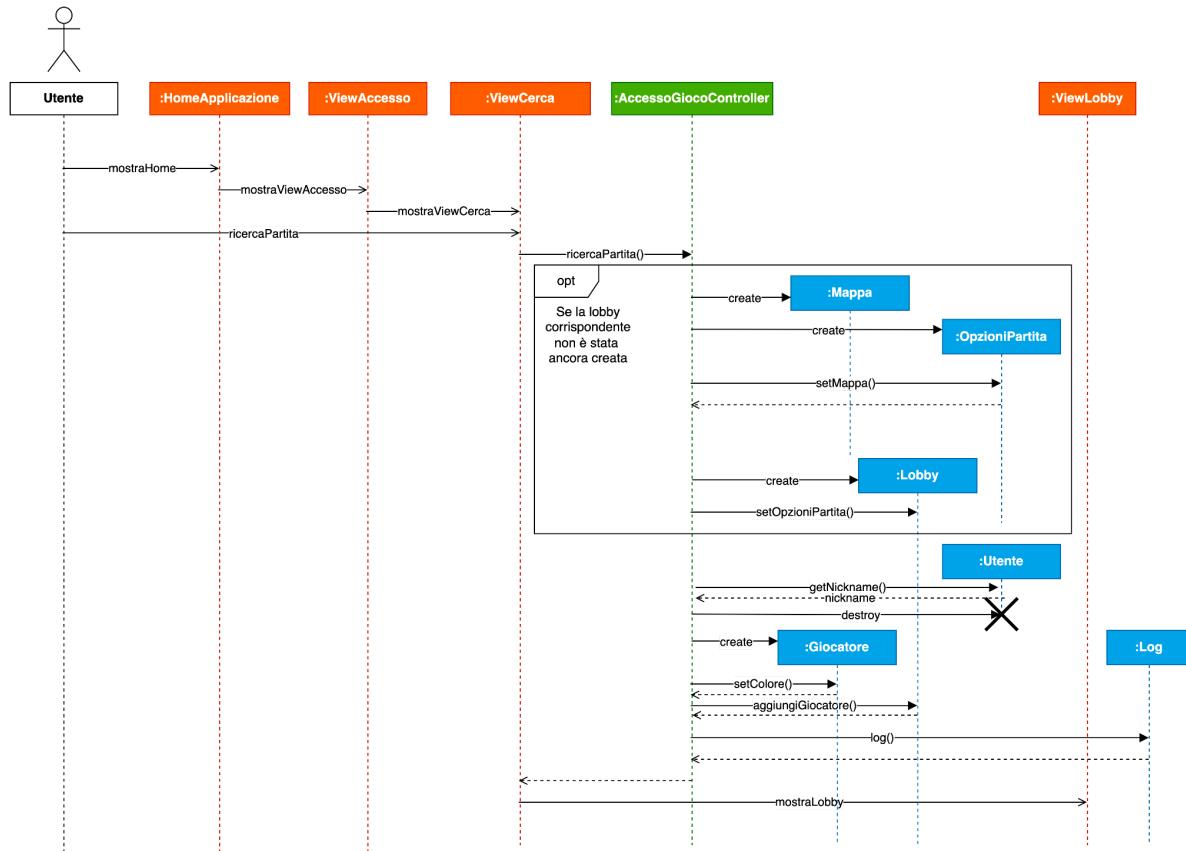


Diagramma di sequenza: Accesso a una partita tramite Ricerca



(1) N.B. prima di distruggere l'Utente ci salviamo il nickname, da poter utilizzare poi nella creazione di un Giocatore. Questa operazione, per semplicità, non verrà ripetuta nelle prossime sequenze

(2) N.B. In questa sequenza e in quelle di creazione ed unione a una partita avviene anche l'assegnazione del colore. Mostriamo solo in questa sequenza tale operazione per motivi di spazio.

Diagramma di sequenza: Accesso a una Partita tramite Creazione

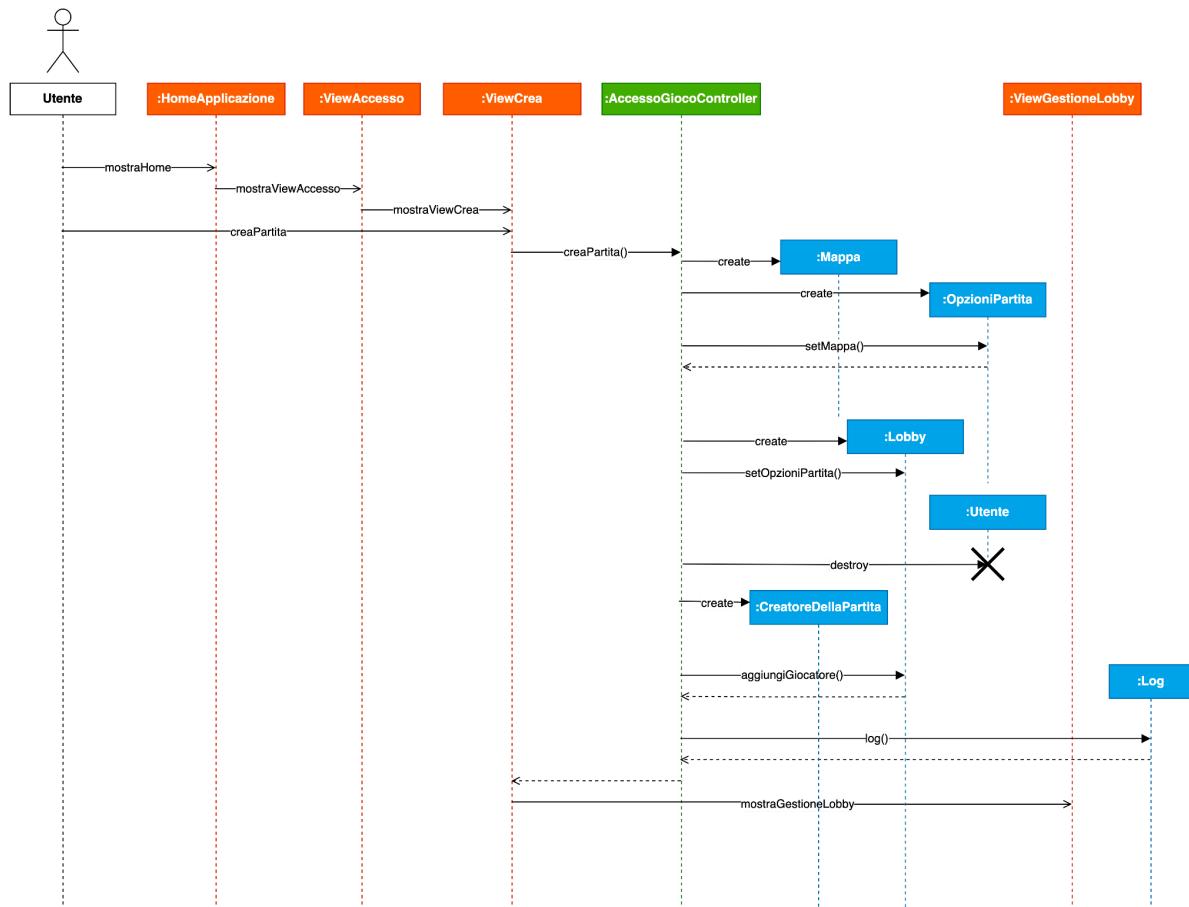


Diagramma di sequenza: Accesso a una Partita tramite Unione

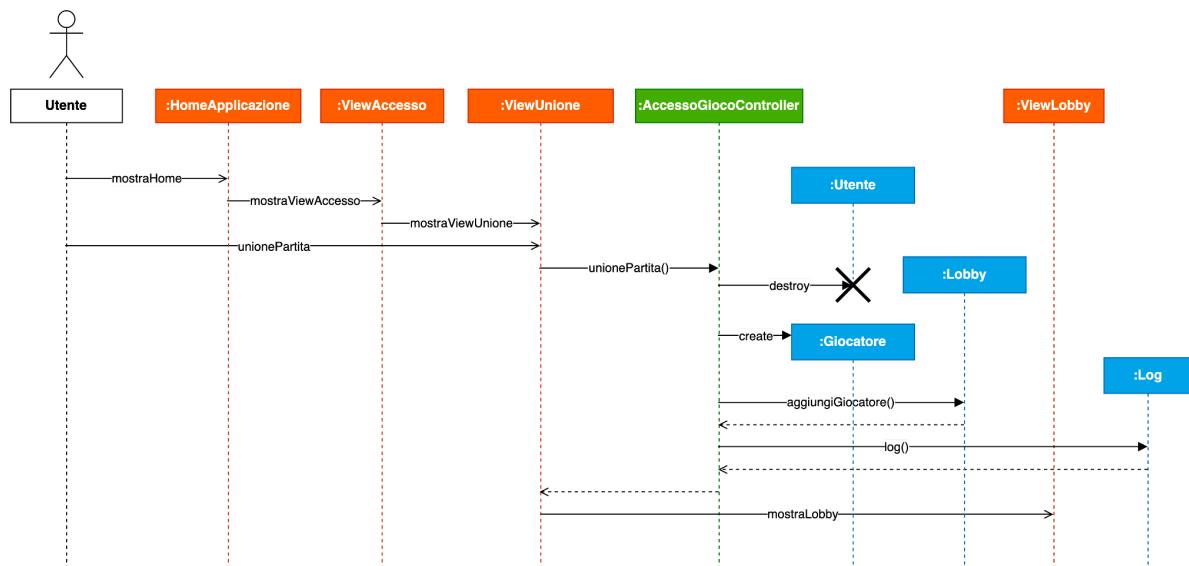


Diagramma di sequenza: Espulsione Giocatore

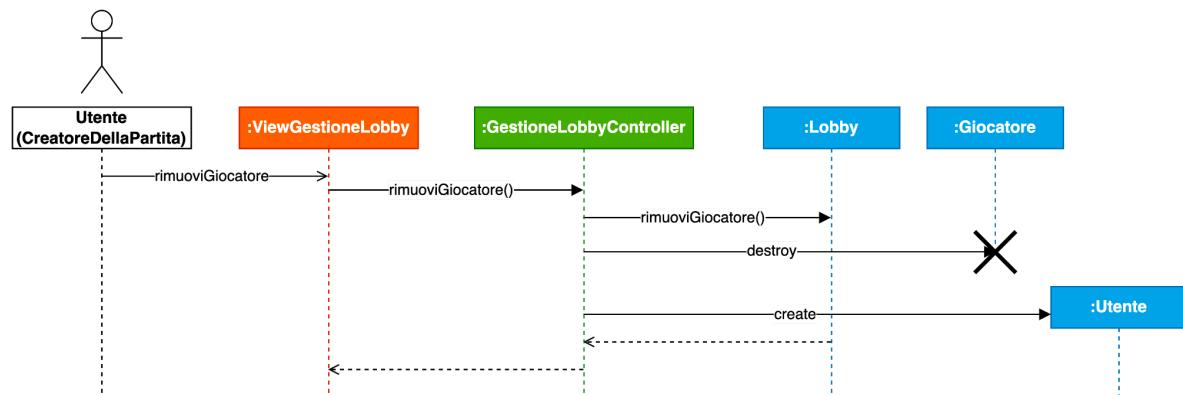


Diagramma di sequenza: Avviamento Partita

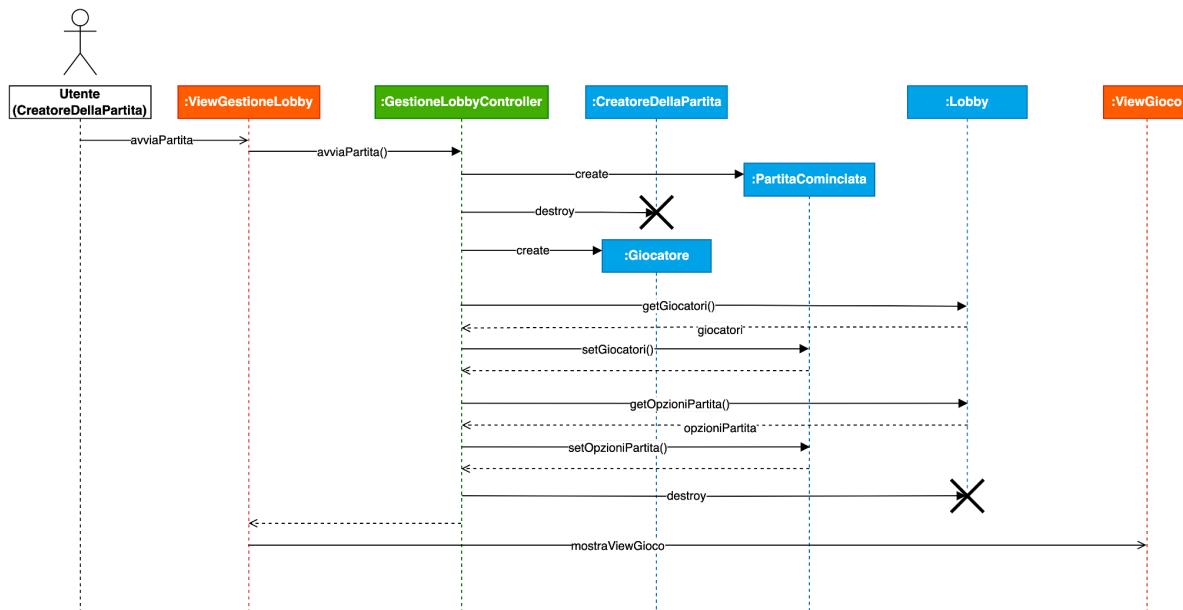


Diagramma di sequenza: Scioglimento Lobby

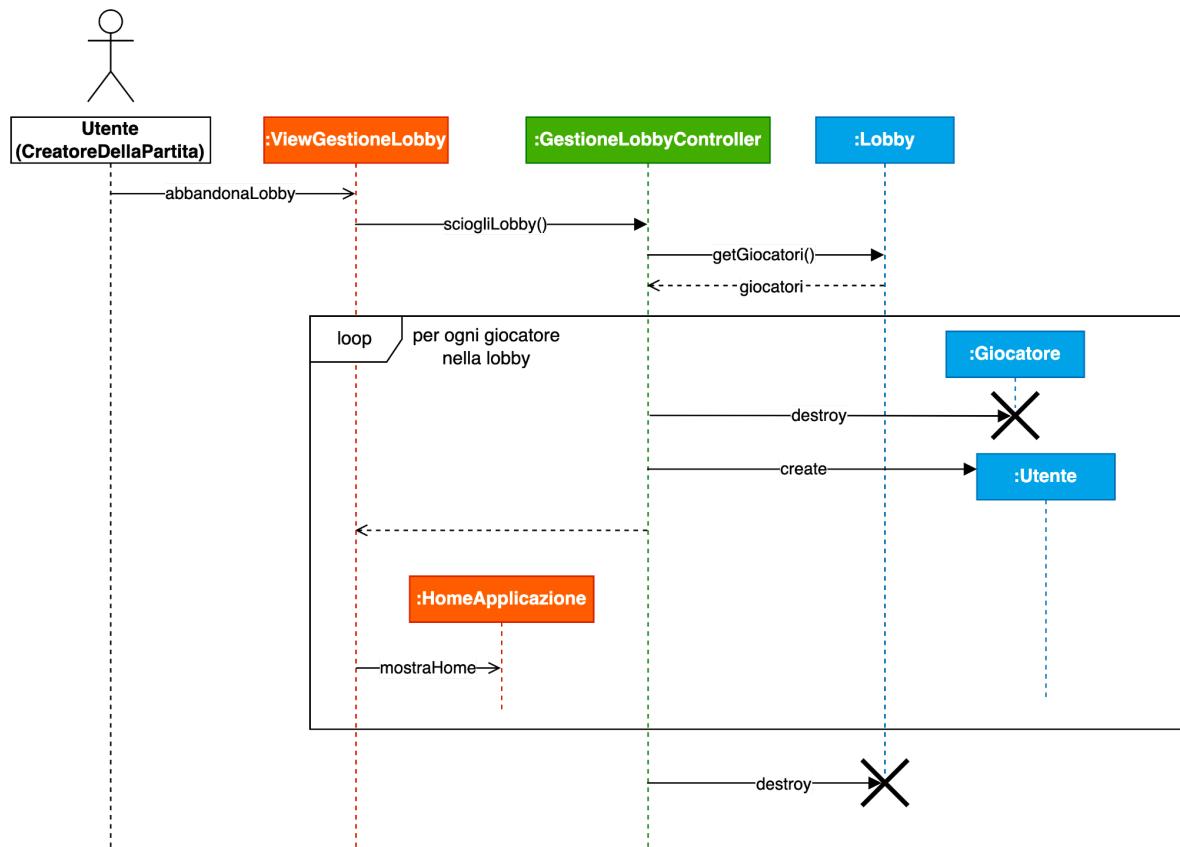
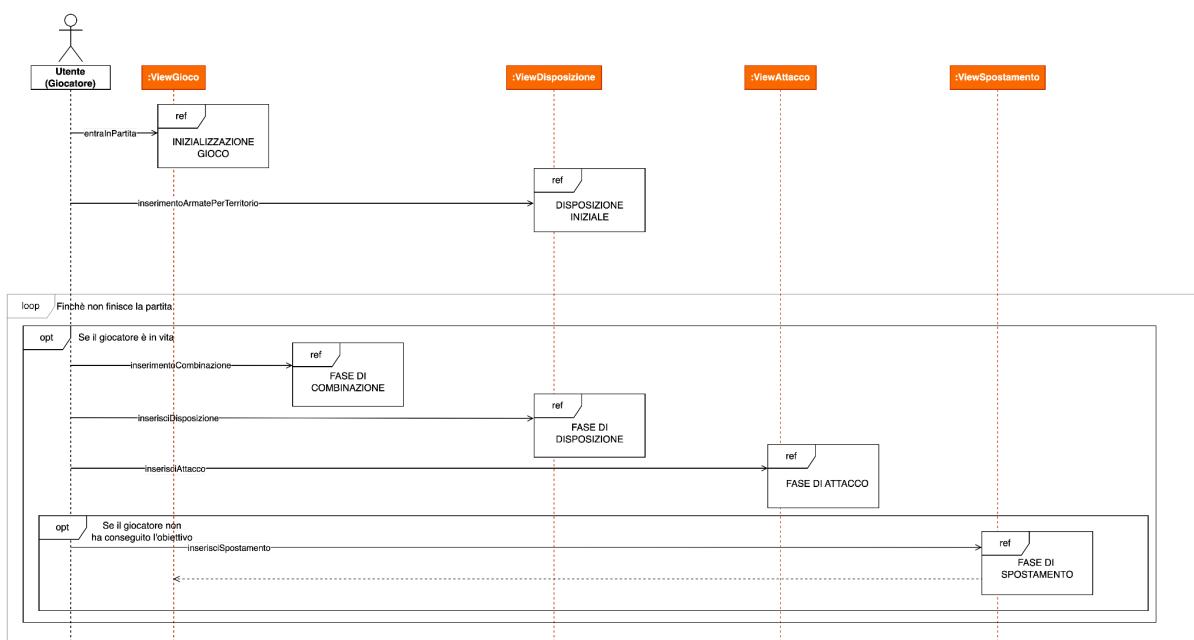
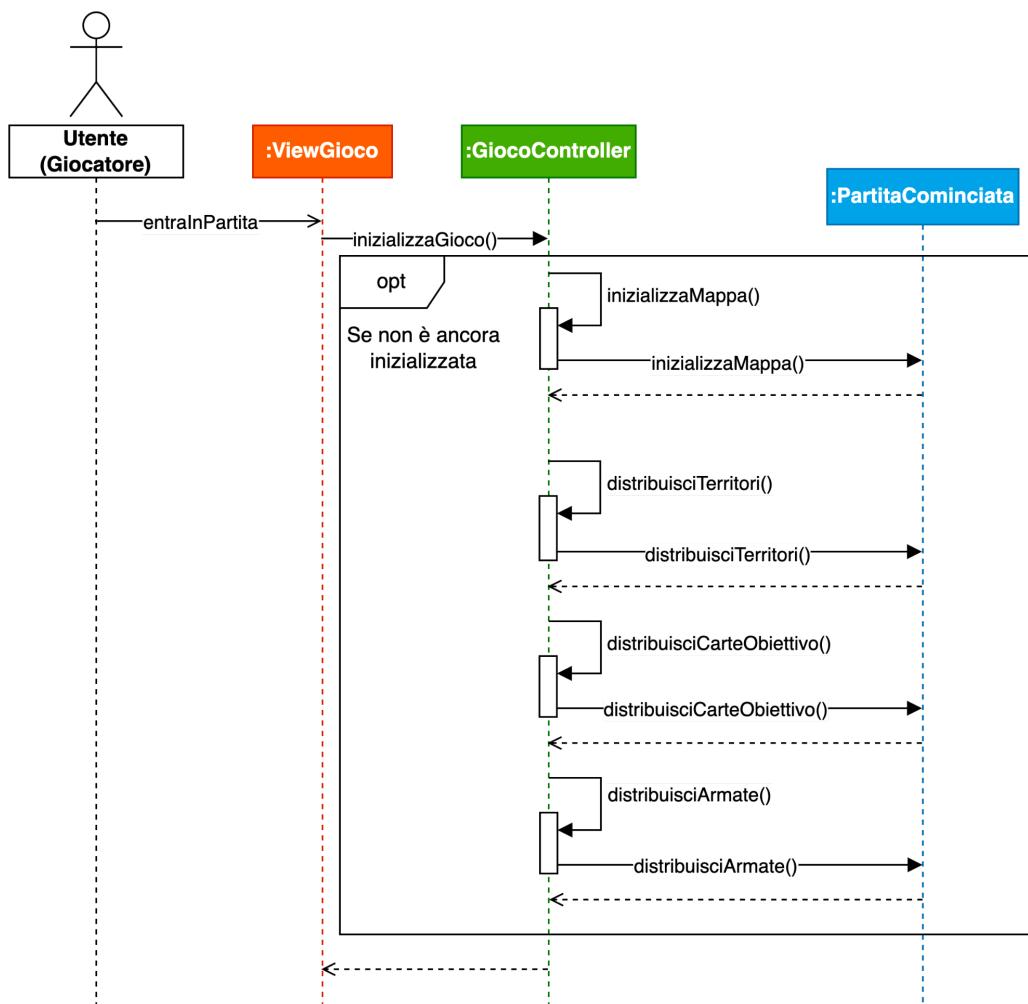


Diagramma di sequenza: Gioca Partita

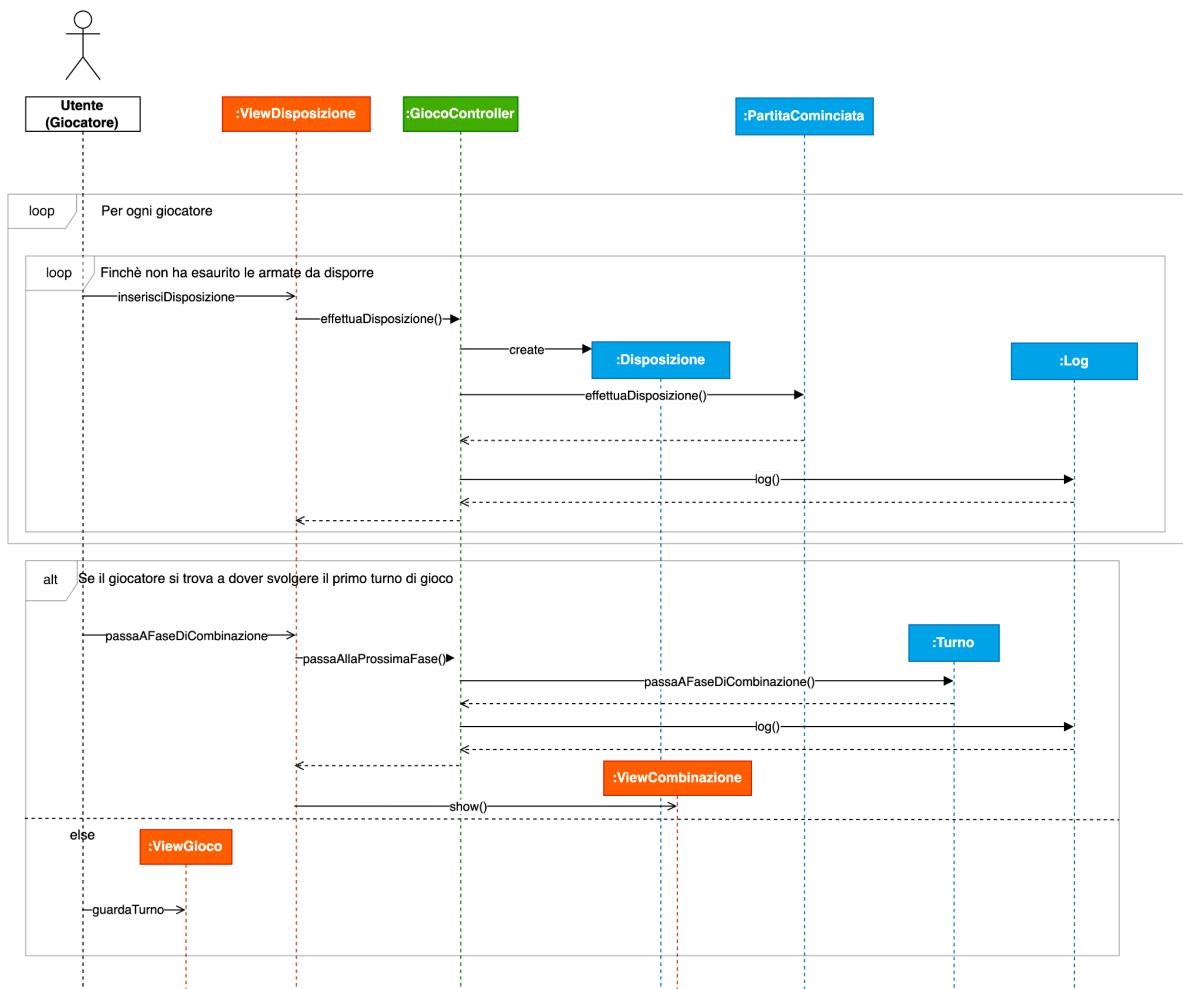
La sequenza di “Gioca Partita” viene di seguito presentata mostrando componenti per motivi di spazio.



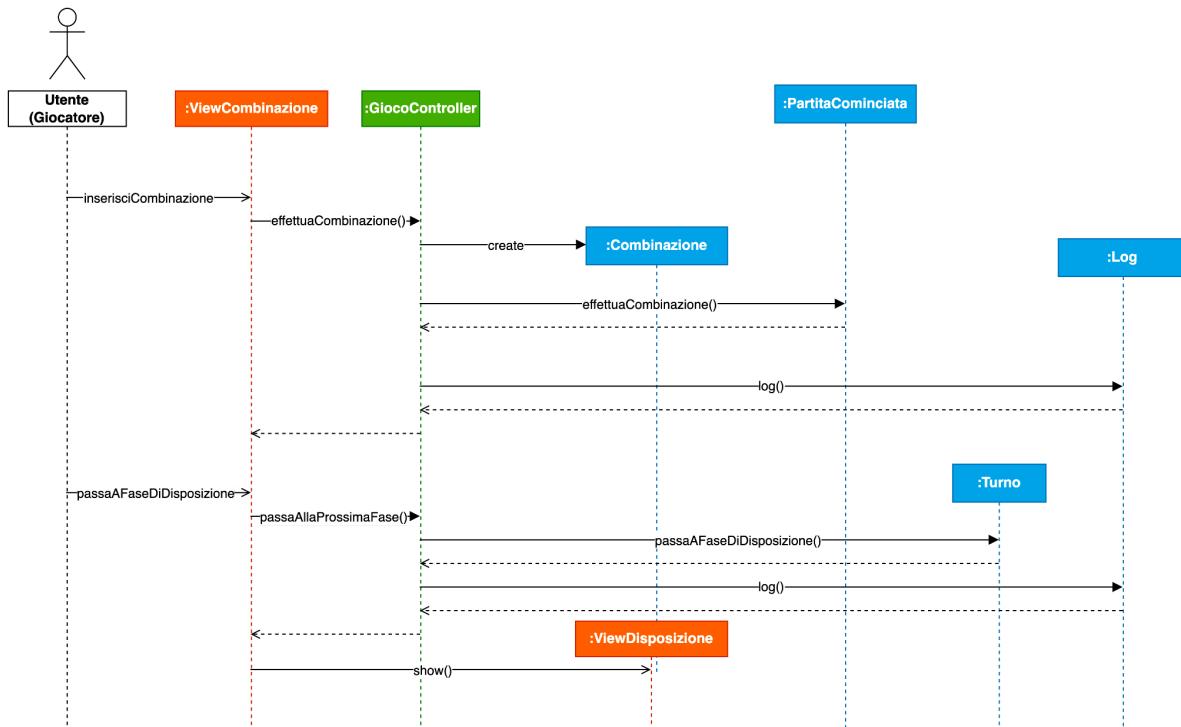
Inizializzazione Gioco



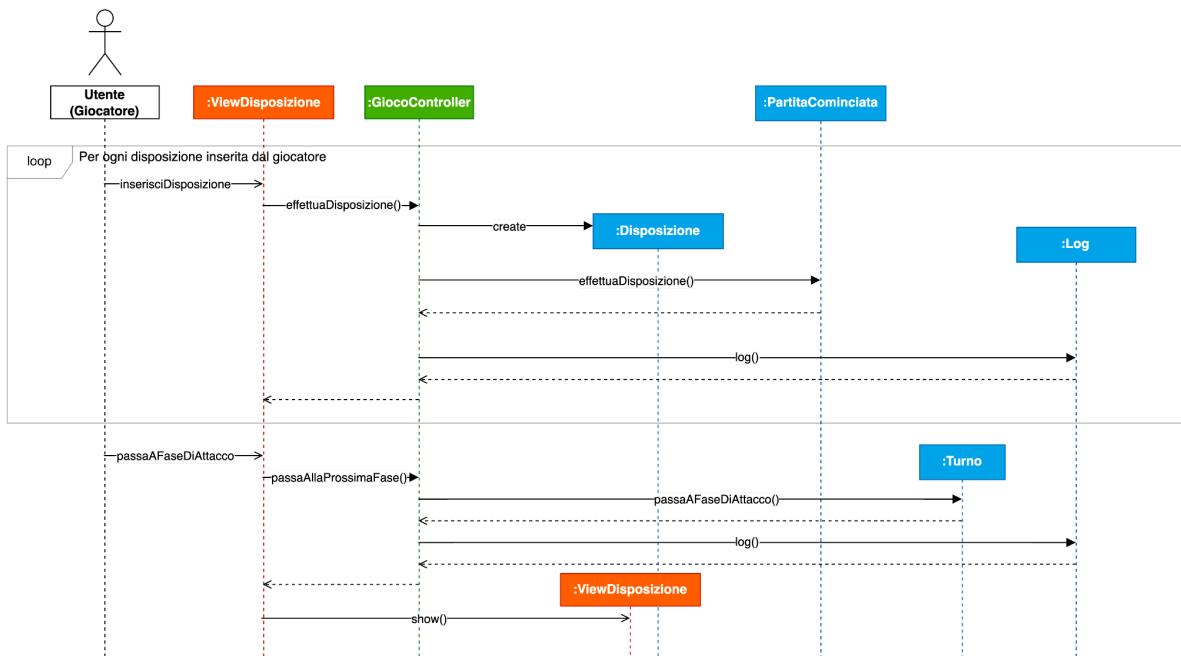
Disposizione Iniziale



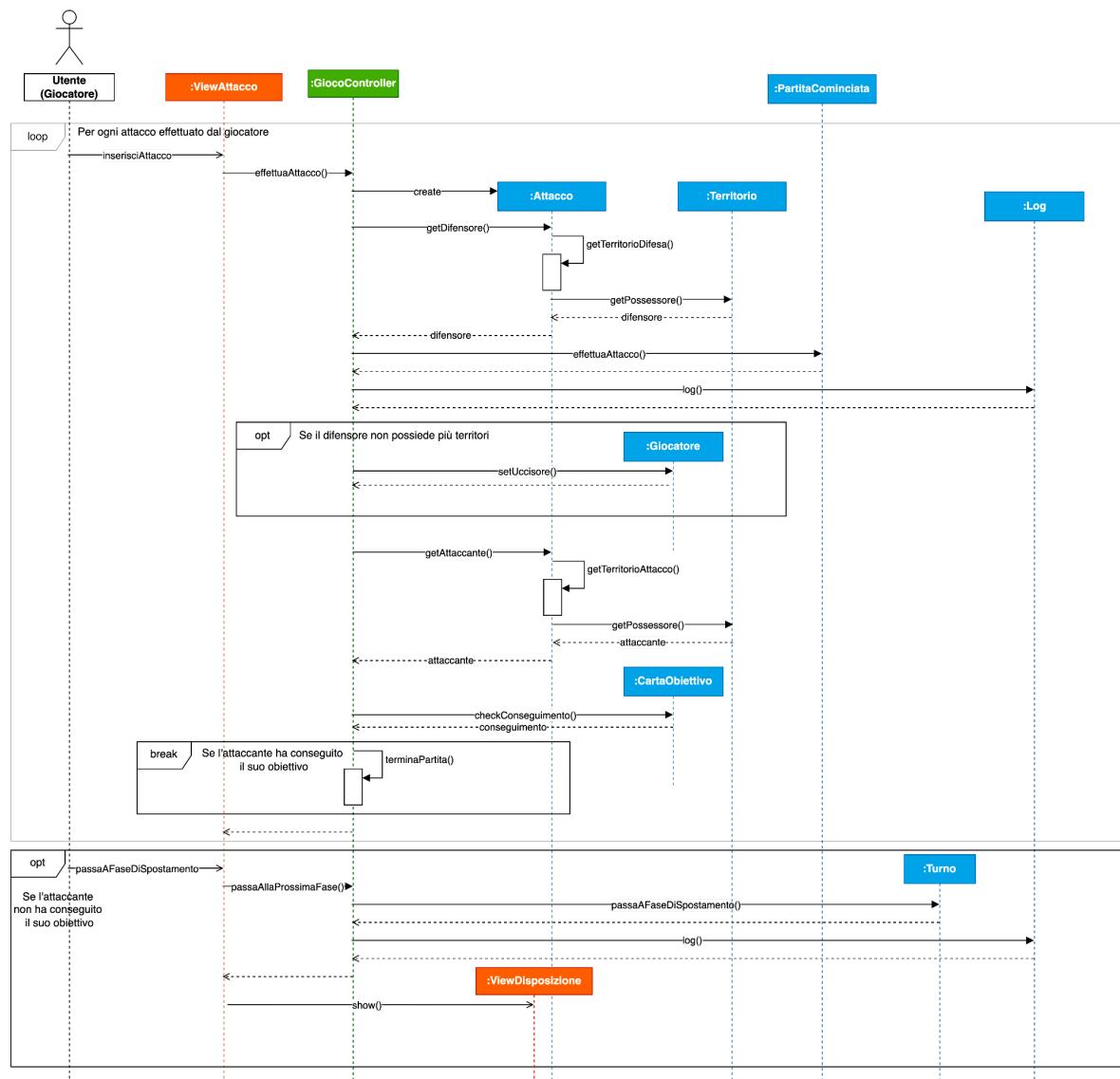
Fase di Combinazione



Fase di Disposizione



Fase di Attacco



Fase di Spostamento

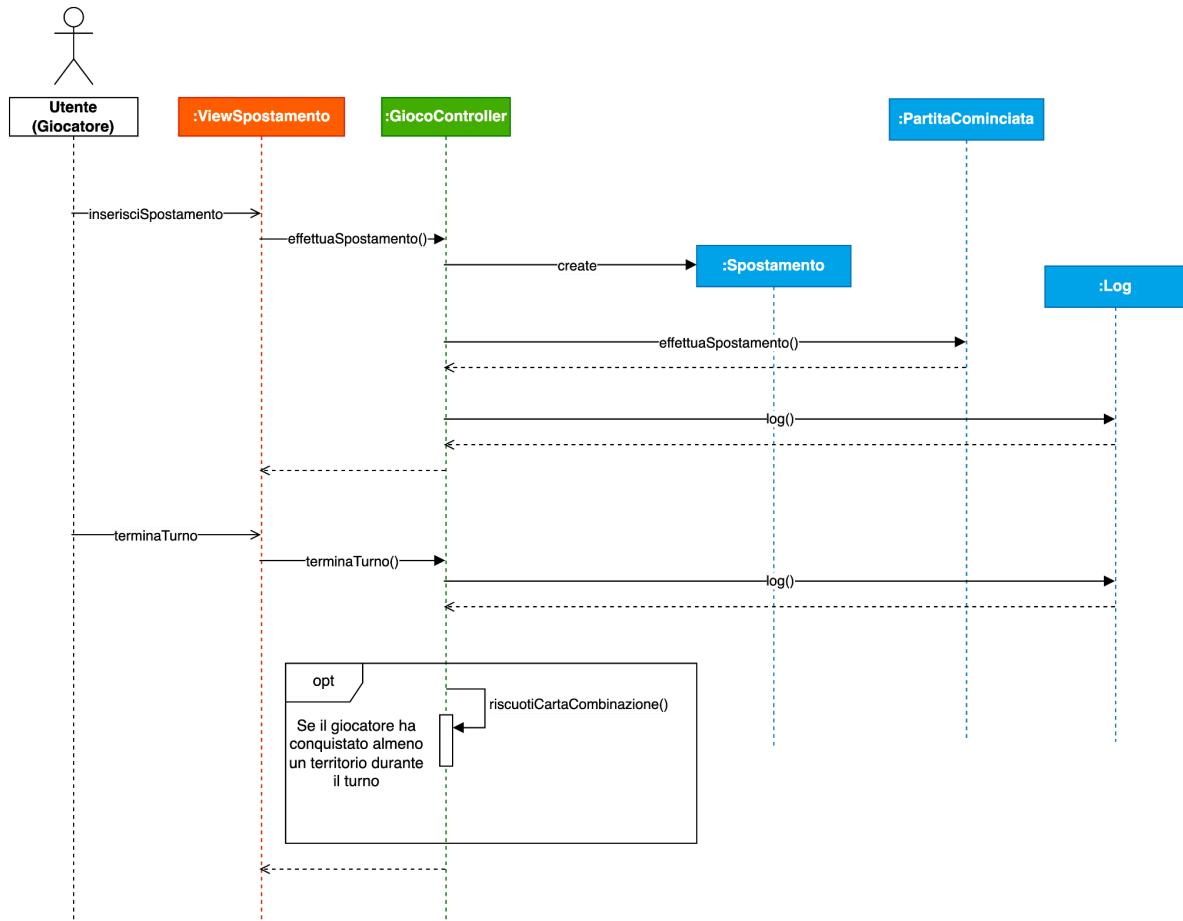
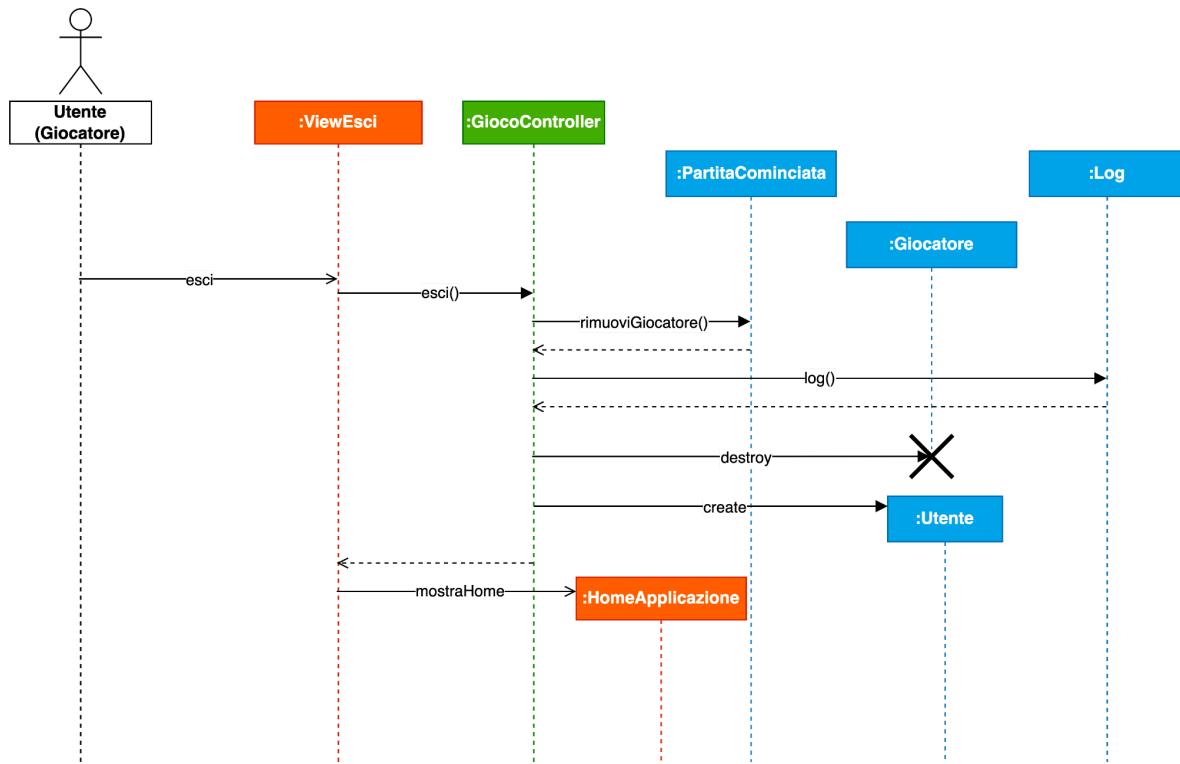
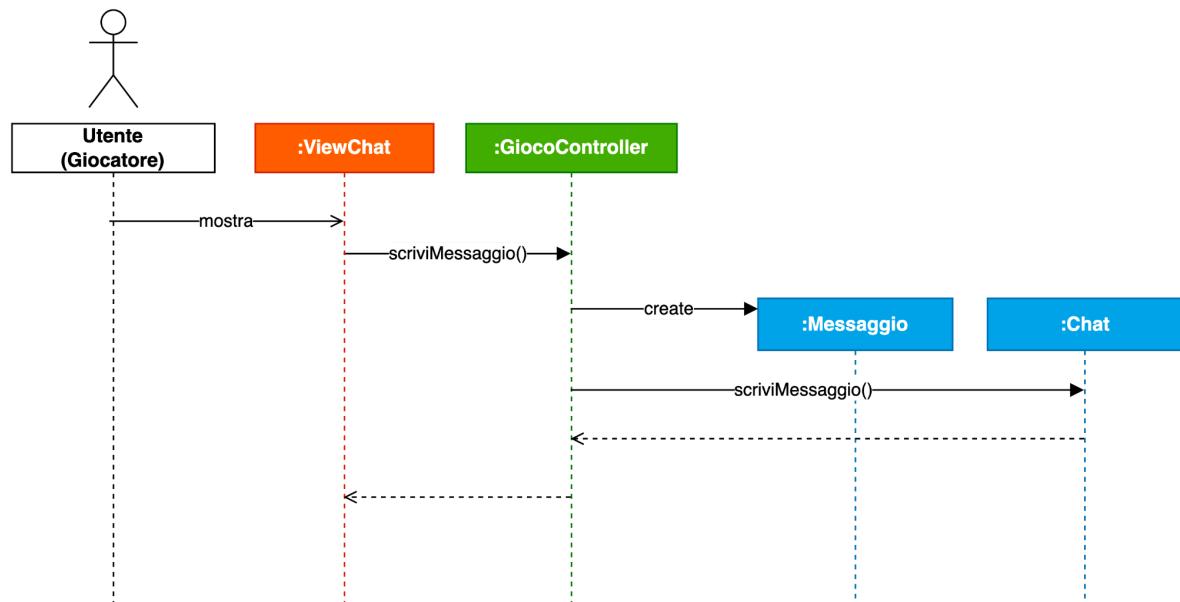


Diagramma di sequenza: Esci dalla partita



N.B. La redistribuzione delle armate e dei territori avviene nella chiamata di *rimuoviGiocatore()*.

Diagramma di sequenza: Utilizzo Chat



Architetture Logica: Comportamento

Diagramma di stato: Stati della Partita

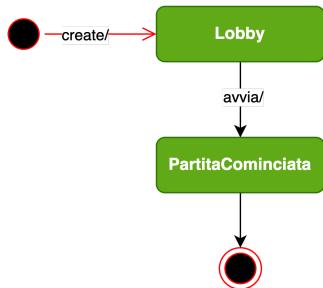
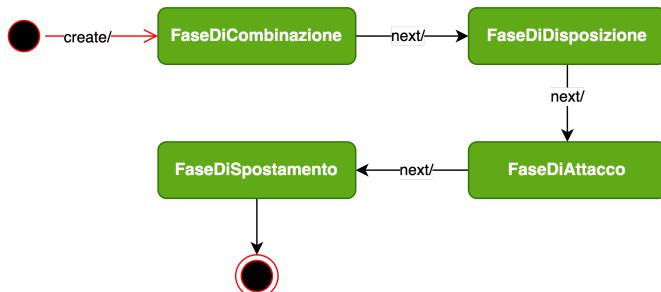


Diagramma di stato: Stati della Fase di un Turno



Piano di lavoro

Il progetto e lo sviluppo del sistema sono assegnati a diversi team come indicato nella tabella sottostante:

Package	Progetto	Sviluppo
Dominio	Jusufi, Pellegrino, Zenere	Pellegrino
ScegliNickname	Jusufi, Pellegrino	Zenere
ConsultaManuale	Jusufi, Zenere	Jusufi
AccessoGioco	Jusufi, Pellegrino, Zenere	Pellegrino
GestionePartita	Jusufi, Pellegrino, Zenere	Zenere
InterfacciaScegliNickname	Pellegrino, Zenere	Jusufi
InterfacciaConsultaManuale	Jusufi, Pellegrino	Pellegrino
InterfacciaAccessoGioco	Pellegrino, Zenere	Zenere
InterfacciaGestionePartita	Jusufi, Pellegrino	Jusufi

Piano del collaudo

A titolo d'esempio si riporta solo il test per la classe **Lobby**.

```
[TestFixture]
public class TestLobby {
    private Lobby _lobby;

    [SetUp]
    public void LobbySetUp() {
        _lobby = new Lobby("678ABC");
    }

    [Test]
    public void Test_Lobby_AggiungiGiocatore() {
        _lobby.aggiungiGiocatore(
            new Giocatore(
                "64d3a532-1e99-4d9b-b79a-a0df10e40549",
                "Pelle"
            )
        );

        var giocatori = _lobby.getGiocatori();

        Assert.That(
            giocatori.Count,
            Is.EqualTo(1);
        )

        var giocatore = giocatori[0];

        Assert.That(
            giocatore.getId(),
            Is.EqualTo("64d3a532-1e99-4d9b-b79a-a0df10e40549")
        );

        Assert.That(
            giocatore.getNickname(),
            Is.EqualTo("Pelle")
        );
    }
}
```

Progettazione

Progettazione Architetturale

Requisiti non funzionali

Nell'Analisi del Problema (Tabella Vincoli) sono emersi due requisiti non funzionali che impongono dei vincoli al sistema:

- Velocità delle comunicazioni
- Protezione delle comunicazioni

Aggiungere strati e meccanismi di cifratura rallenterebbe lo scambio di informazioni tra server e client, facendo perdere all'applicativo l'obiettivo di fornire l'esperienza in tempo reale di una partita di Risiko. L'attacco al sistema può avere diversi effetti, da un peggioramento dell'esperienza di gioco a una manomissione delle funzionalità del server.

Scelta dell'architettura

Dal punto di vista architettonico, la scelta più idonea per questo tipo di sistema è un'architettura client/server a due livelli.

L1-Client:

Si è deciso di sviluppare un solo client che contenga entrambe le funzionalità di accesso gioco e gioco effettivo di una partita.

L2-Server:

Per quel che riguarda il lato server, si è deciso di adottare server diversi per la gestione delle diverse funzionalità:

- Un server http per la funzionalità di accesso gioco
- Uno o più server websocket per la gestione di ciascuna Lobby
- Un server http per la funzionalità di creazione delle partite
- Uno o più server websocket per la gestione di ciascuna Partita
- Un server per la funzionalità di gestione dei Log

Si è deciso di separare la funzionalità di accesso gioco e di gestione della partita, poiché offrono un insieme di funzionalità completamente differenti.

Sarà, infatti, presente un server http che gestirà l'accesso a una partita e che gestirà i vari server websocket creati, i quali sono associati a una Lobby, che permetteranno la comunicazione in tempo reale di avvenimenti (esempio: espulsione di un giocatore).

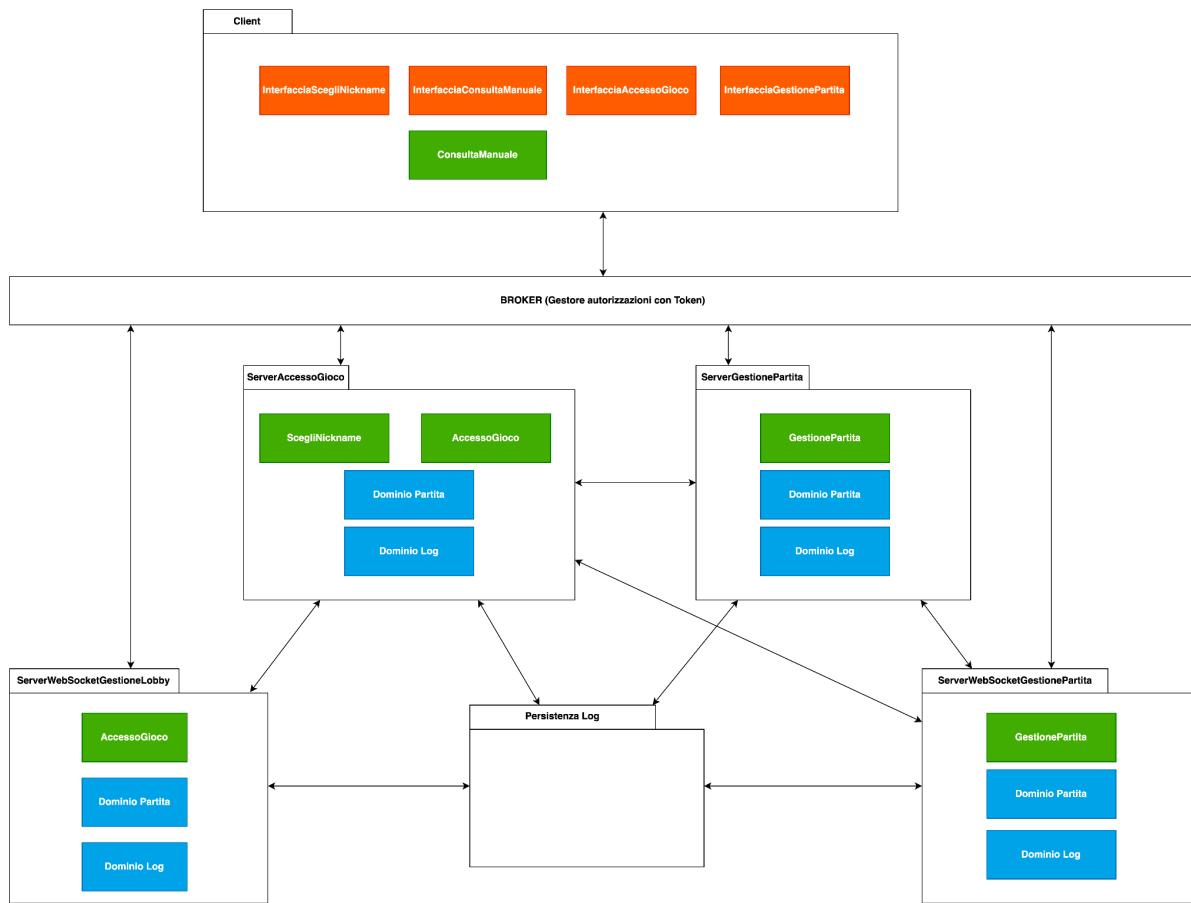
Per quanto riguarda la gestione delle partite, invece, sarà presente un server http che si occuperà di creare server websocket e di associarli a una partita cominciata. Inoltre, ciascun server websocket verrà avviato all'interno di un thread separato.

Si è inoltre deciso di adottare il Pattern Broker per la gestione della sessione e per proteggere ulteriormente lo strato dei server. Tale scelta inoltre permette di applicare il Dependency Inversion Principle: disaccoppiando i client dai server, la struttura dei server è "nascosta" dal broker e diventa abbastanza semplice "sostituire" un server oppure aggiungerne uno nuovo (design for change).

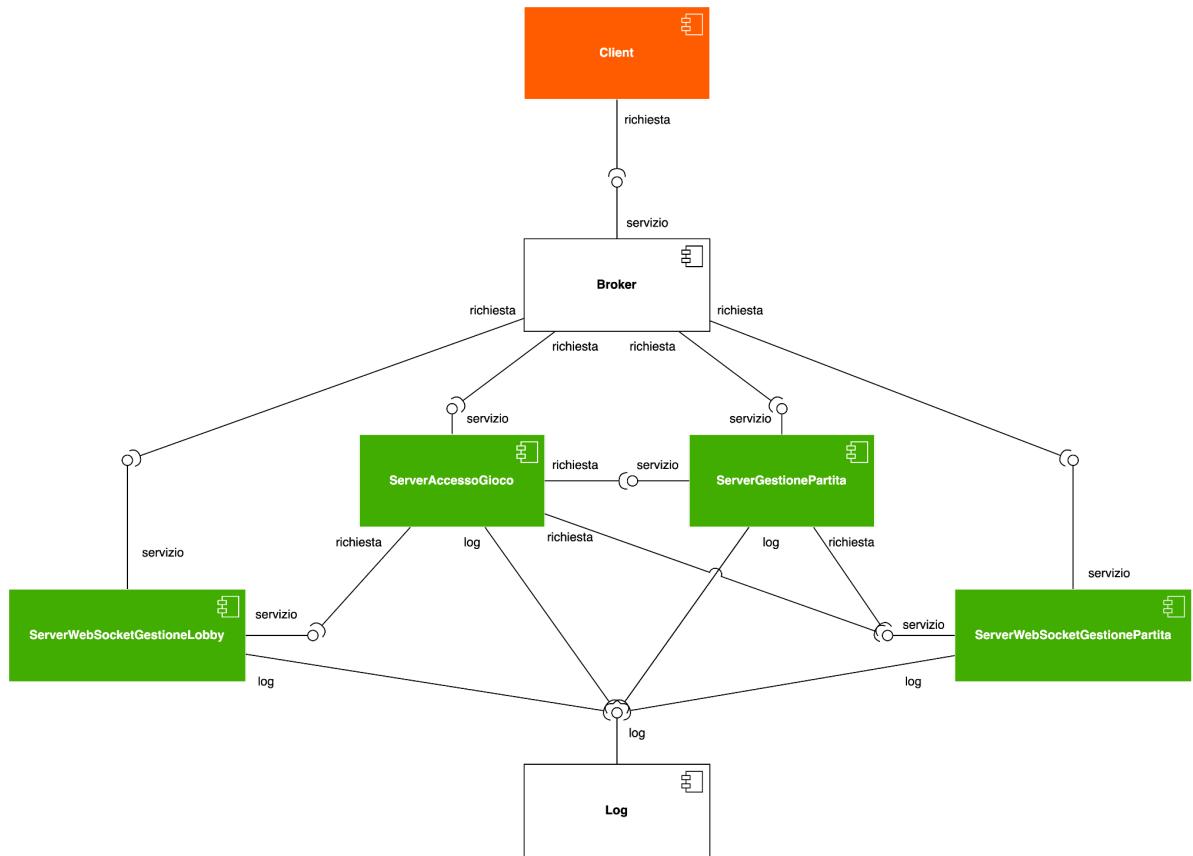
Infine, per garantire la sicurezza nella comunicazione client/server si è deciso di utilizzare il protocollo TLS (transport layer security). TLS è un protocollo crittografico che permette una comunicazione sicura dalla sorgente al destinatario (end-to-end) su reti TCP/IP (come ad esempio, Internet) fornendo autenticazione, integrità dei dati e cifratura, operando al di sopra

del livello di trasporto. TLS si può usare sia per lo scambio di messaggi http (utilizzando il protocollo https), sia per lo scambio di messaggi websocket (utilizzando il protocollo wss).

Nella figura sottostante è riportata l'architettura del sistema organizzata attraverso un diagramma dei package.



Nella figura sottostante è riportata l'Architettura del Sistema organizzata attraverso un diagramma dei componenti.



Scelta tecnologica

Per il client si è deciso di utilizzare la libreria *React.js*, che permette di creare facilmente interfacce grafiche con logiche complesse.

Per il server di accesso gioco e di gestione partita si è deciso di utilizzare *node.js*, un runtime di javascript, grazie al quale si possono creare sia server http sia server websocket.

Per motivi dovuti alla vastità del progetto, abbiamo constatato che utilizzare un linguaggio non tipato come javascript non sarebbe però stata una scelta lungimirante, dunque per non rinunciare all'ambiente di javascript e tutte le tecnologie annesse (*node.js* e librerie come *react.js*) abbiamo deciso di utilizzare *TypeScript* come linguaggio di programmazione, che è un superset di javascript che ammette i tipi.

Queste scelte tecnologiche ci portano a mettere a disposizione un server Host per la Build del progetto React, invece che installare il Client direttamente nella macchina del cliente.

Nei diagrammi relativi al deployment mostriamo come il cliente deve recuperare, tramite un browser, la webapp dal server host della build del front-end.

Per lo scambio di dati si userà il formato JSON.

Progettazione di dettaglio

Nel seguito si riportano i diagrammi di dettaglio delle varie parti del Sistema.

Struttura

Di seguito riportiamo i modelli di dominio.

Per semplificare la lettura abbiamo scelto di suddividerli in più parti e utilizzando referenze di classi vuote per descrivere relazioni tra classi non appartenenti alla stessa sezione.

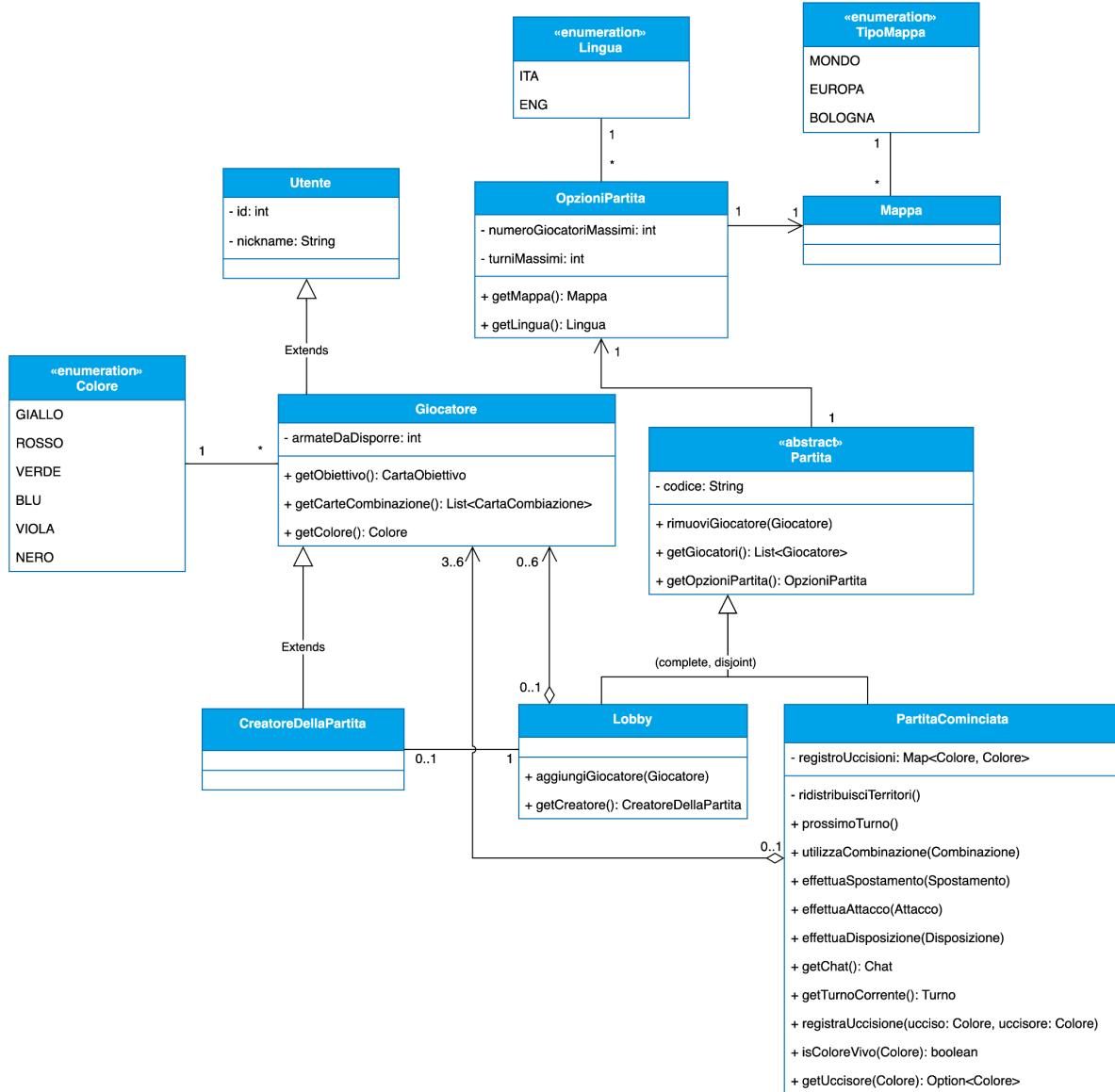
Esempio: la classe *PartitaCominciata* è mostrata nel dettaglio nella sezione dedicata alla definizione generica della partita e viene riportata senza metodi e attributi nella sezione di strutturazione dei Turni e delle Fasi.

Diagramma di Dettaglio: Dominio-Partita

Il seguente diagramma delle classi rappresenta la parte del modello del dominio relativa alla partita e alle sue componenti.

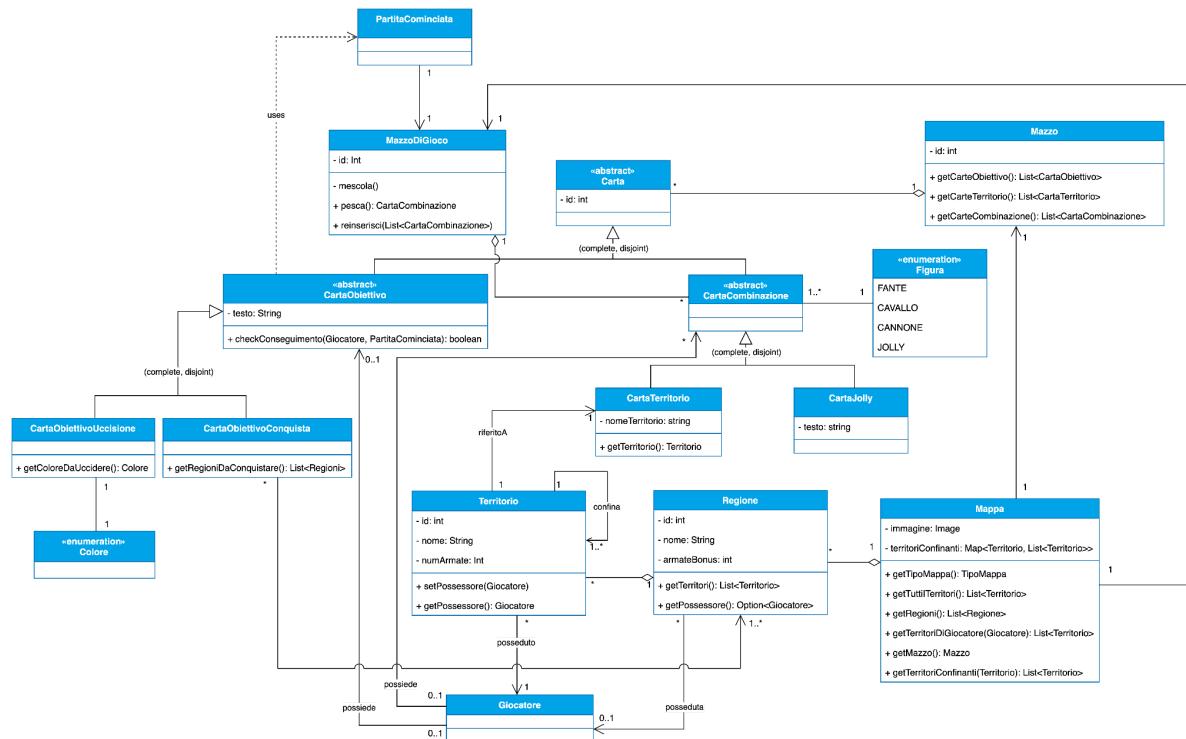
Per semplificare la lettura abbiamo suddiviso il dominio della Partita in 3 parti dove le relazioni e gli attributi, una volta definiti, non verranno riportati nei ritagli successivi:

1) Definizione generica della partita



Come scelta progettuale è stato deciso di delegare la registrazione delle uccisioni alla **PartitaCominciata** tramite `registroUccisioni`, poiché così le informazioni sono più facilmente recuperabile durante il controllo del conseguimento dell'obiettivo e al tempo stesso si può determinare più facilmente se il giocatore sia vivo o morto durante la creazione di un nuovo turno.

2) Gli elementi presenti nella partita



Come scelta progettuale abbiamo suddiviso *CartaObiettivo* in *CartaObiettivoUccisione* e *CartaObiettivoConquista* utilizzando il *Pattern Strategy* per avere un codice appropriato quando facciamo il controllo dell'obiettivo, dividendolo in 2 macrocategorie, dato che queste richiedono operazioni diverse.

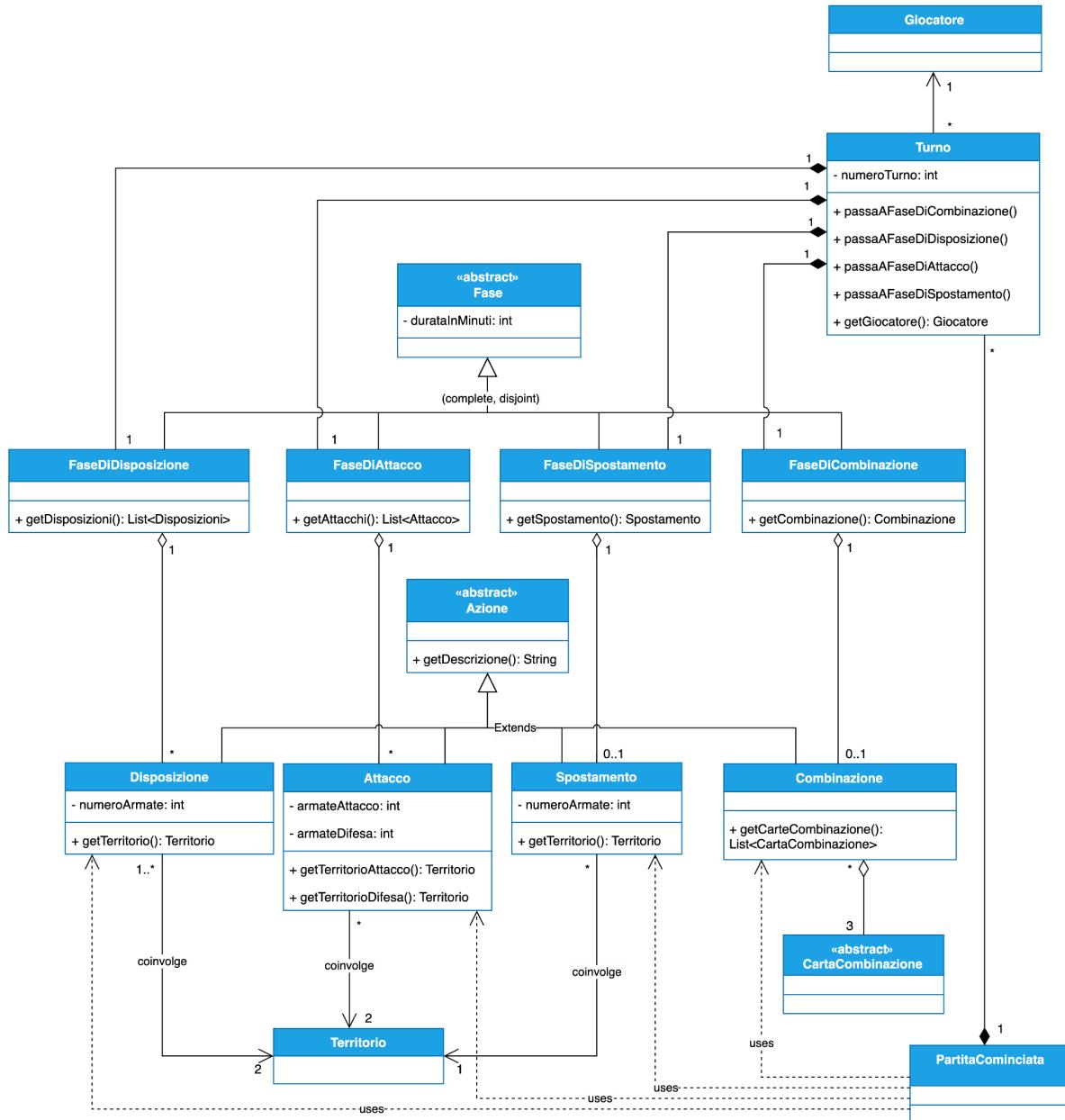
Inoltre abbiamo risolto la relazione di confinanza tra territori aggiungendo un grafo dentro la classe *Mappa*.

La classe *Mappa*, avendo un istanza della lista delle *Regioni*, che a loro volta hanno una lista dei *Territori* a loro associati, riesce a recuperare tutti i *Territori* e calcolare quali siano i *Territori* posseduti da un certo *Giocatore*.

La relazione tra *Territorio* e *CartaTerritorio* la risolviamo mettendo un'istanza di quest'ultima dentro la prima; quindi se, ad esempio, ho bisogno di controllare durante una combinazione se le carte territorio giocate sono riferite a territori che il giocatore possiede, in modo da ottenere il bonus, primo chiamo *mappa.getTerritoriDiGiocatore()*, poi con un ciclo controllo se la carta territorio è associata a uno di questi territori.

Per quanto riguarda *Mazzo* abbiamo deciso di dividere le funzionalità di recupero delle carte da quelle di effettivo gioco. Le funzionalità di gioco vengono gestite da *MazzoDiGioco*, il quale, una volta ricevute le *CarteCombinazione* recuperate da *Mazzo*, gestirà il pescaggio e il reinserimento delle carte dopo un'utilizzazione di una combinazione.

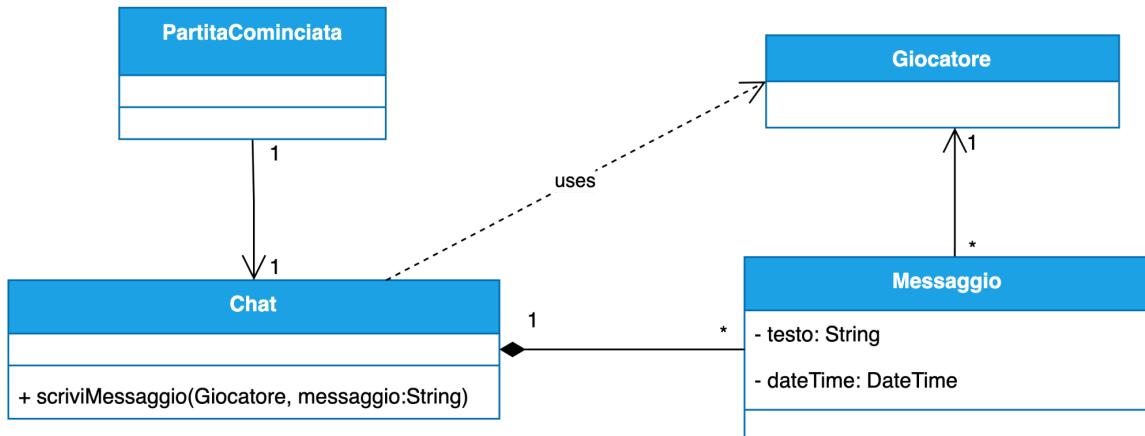
3) La strutturazione dei Turni e delle Fasi



Come scelta progettuale abbiamo aggiunto **Azione** utilizzando il *Pattern Strategy* così che ogni classe che implementa azione avrà il metodo *getDescrizione()* che permette di ottenere la descrizione di quello che è avvenuto.

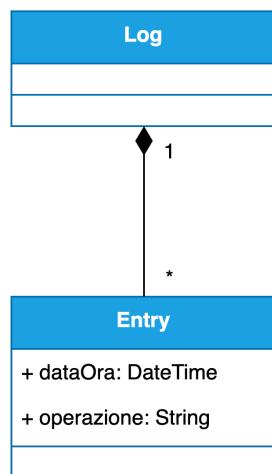
Inoltre abbiamo risolto la composizione tra **PartitaCominciata** e **Turno** utilizzando un'istanza di *Stream<Turno>*; la callback passata al costruttore dello Stream specifica le regole con la quale viene generato un nuovo turno, aumentando progressivamente il numero del turno e tenendo in considerazione i giocatori morti. Il metodo *next()* dello Stream ci dà l'istanza del turno, che salviamo nella proprietà *turnoCorrente* di **PartitaCominciata**.

4) La chat



Per quanto riguarda la gestione della chat, non sono stati effettuati cambiamenti rispetto all'analisi.

Diagramma di Dettaglio: Dominio-Log



Per quanto riguarda la gestione dei log, è stato deciso di delegare le richieste di scrittura sul log fisico alla classe Log dopo la creazione di una Entry.

Diagramma di Dettaglio: Interfacce nel Client

Consultazione del manuale

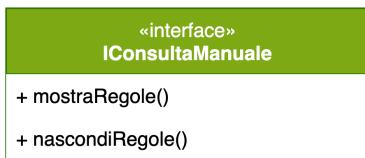


Diagramma di Dettaglio: Interfacce nei Server

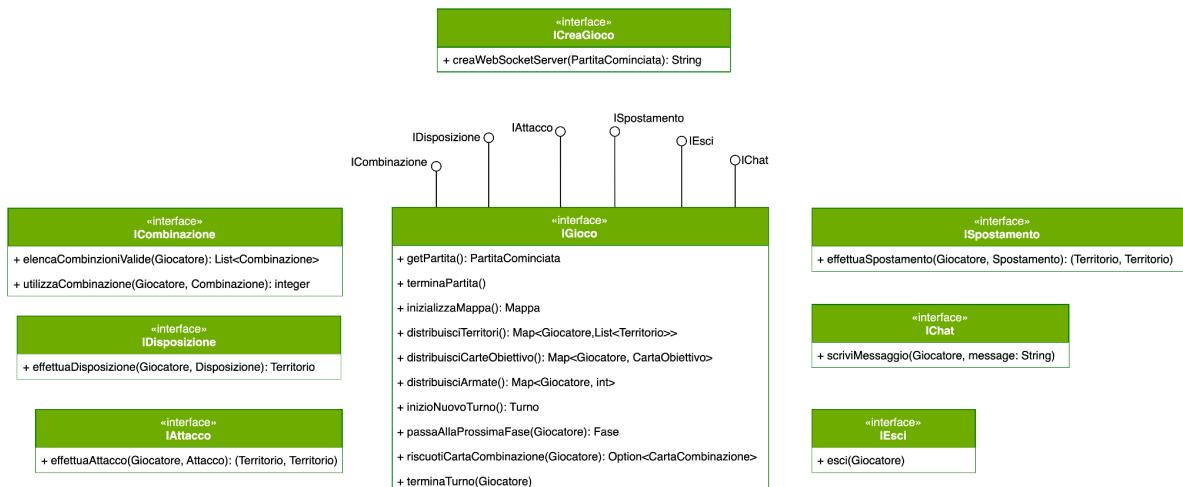
Scelta del nickname



Accesso al gioco



Gestione della partita



Le interfacce permettono di applicare *The Dependency Inversion Principle* ovvero i moduli di alto livello non dipenderanno direttamente dai moduli di basso livello ma da astrazioni costanti nel tempo.

Diagramma di Dettaglio: ScegliNickname

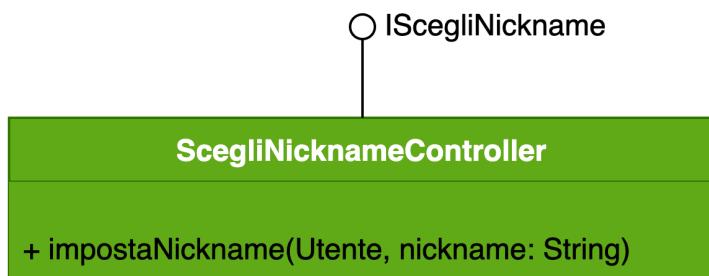


Diagramma di Dettaglio: ConsultaManuale

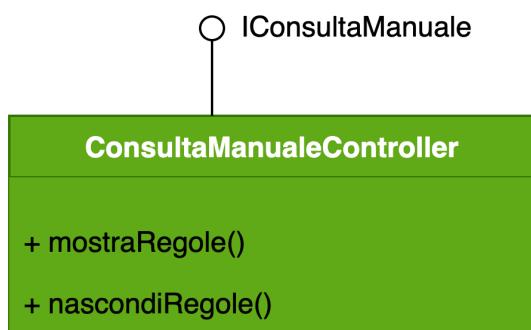
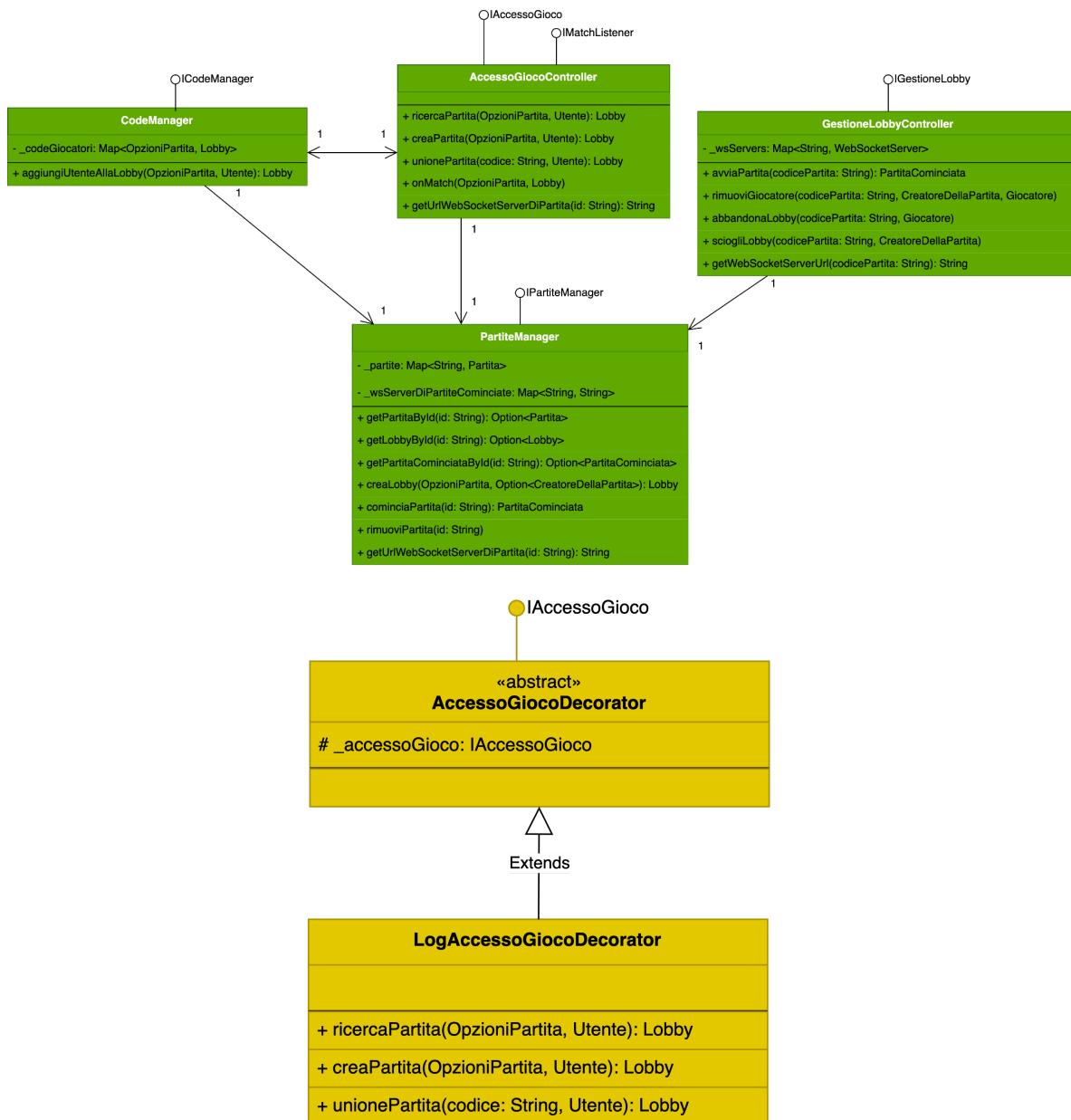


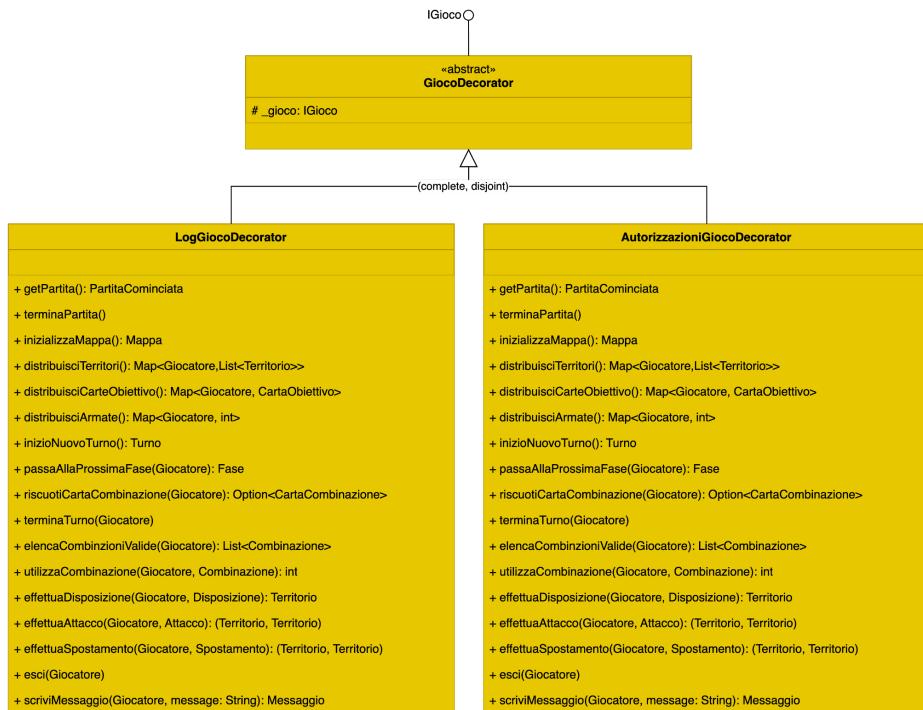
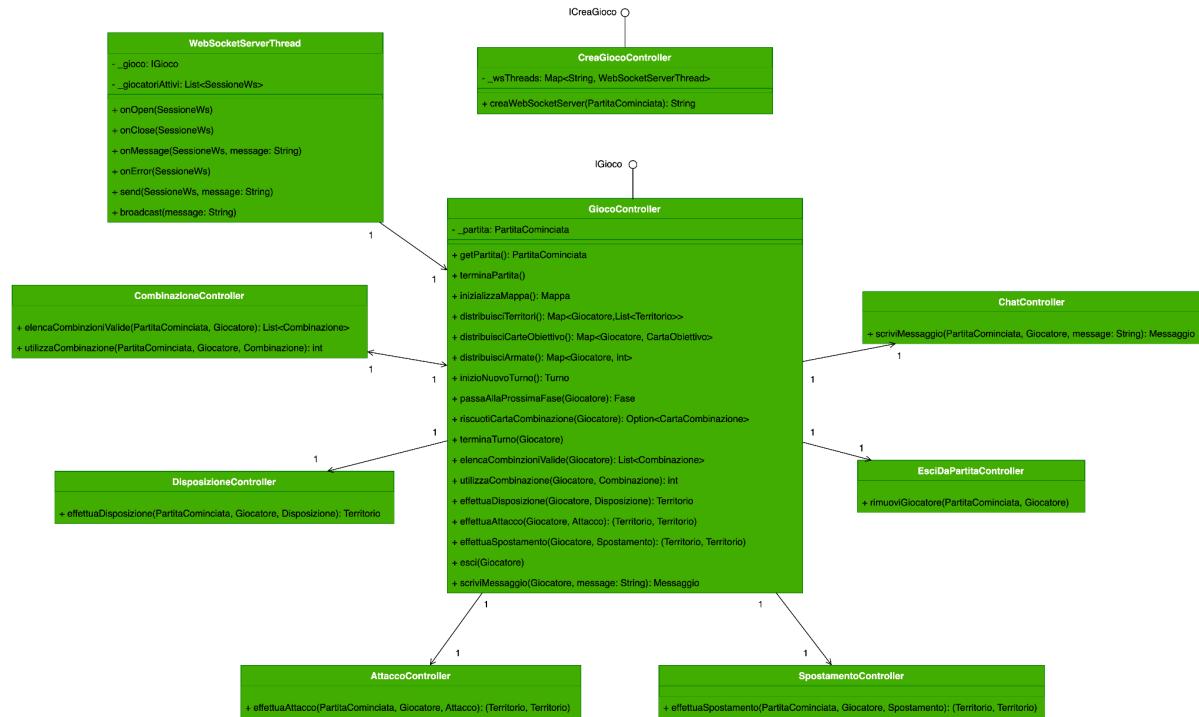
Diagramma di Dettaglio: AccessoGioco



Si è deciso di creare 2 classi di utilità: PartiteManager e CodeManager.
 PartiteManager si occupa della gestione di recupero e creazione delle partite.
 CodeManager si occupa di combinare i vari utenti che vogliono partecipare a una Partita con le stesse Opzioni.
 Per avviare una partita viene invocato il metodo cominciaPartita() di PartiteManager, che manda una richiesta http al server di GestionePartite, il quale ritorna l'url sul quale è stato aperto il server WebSocket associato alla Partita.
 Per gestire la connessione WebSocket alle PartiteCominciate, dopo l'avvio di quest'ultime, l'utente deve chiamare getUrlWebSocketServerDiPartita() che fornisce l'url del server WebSocket a cui un Giocatore si deve collegare.
 Invece per gestire il collegamento ai server WebSocket relativi alle Lobby, GestionePartiteController associa a ogni codice Lobby un server WebSocket, il quale condivide in tempo reale gli avvenimenti.

Inoltre, si è deciso di utilizzare il *Pattern Decorator* per rispettare il *Single Responsibility Principle*, in modo da poter eseguire il log delle varie operazioni senza scriverle direttamente dentro ai metodi dei controller. Anche se non è mostrato, usiamo questo pattern anche per le funzionalità di GestioneLobby e ScegliNickname.

Diagramma di Dettaglio: GestionePartita



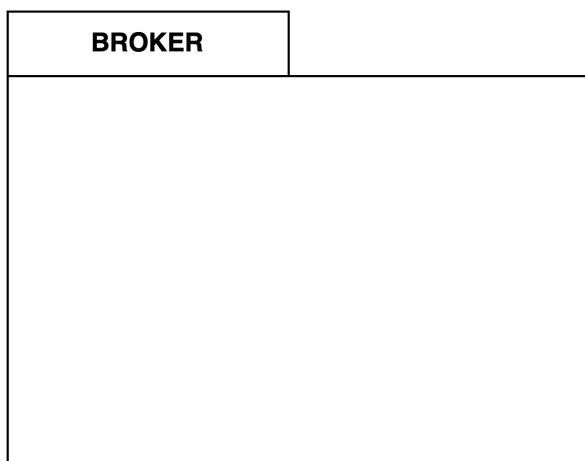
Come scelta progettuale è stato deciso di aggiungere 2 classi di utilità, ovvero *CreaGiocoController* e *WebSocketServerThread*. *CreaGiocoController* si occupa della creazione di server WebSocket ed associare al codice di una *PartitaCominciata* a un'istanza di *WebSocketServerThread*. La classe *WebSocketServerThread* implementa *Runnable* ed estende *WebSocketServer* in modo da poter eseguire su un thread separato e allo stesso tempo di poter offrire le funzionalità di un server WebSocket standard. Si è deciso inoltre che il parsing dei parametri verrà fatto dentro al metodo *onMessage()*.

E' stato deciso, inoltre, di riproporre il *Pattern Decorator* aggiungendo un Decorator concreto in più rispetto a quanto mostrato in precedenza, ovvero quello delle autorizzazioni, che si occuperà principalmente di garantire che certe operazioni svolte durante la partita siano svolte solo da certi Giocatori e solo in certi momenti (ad esempio un Giocatore può svolgere una combinazione solo se si trova nel suo turno e solo se è nella fase di combinazione). L'istanza di *IGioco* in *WebSocketServerThread* sarà un'istanza di *LogGiocoDecorator*, che contiene un *AutorizzazioniGiocoDecorator*, che contiene a sua volta un'istanza *GiocoController* (il componente concreto). E' stata fatta questa scelta perché vogliamo eseguire il log anche di operazioni non autorizzate, in modo che un admin di sistema possa analizzare i file di log prodotti per trovare potenziali problemi.

Da notare come molti metodi, rispetto all'analisi, sono diventati pubblici, in modo che *WebSocketServerThread* possa chiamare quei servizi e gestire i valori di ritorno da condividere tra i client in broadcast. Si noti anche che per non generare traffico inutile si notificano solo i cambiamenti di una partita e non si ritorna ai clienti l'intero stato della partita (ad esempio in un attacco viene ritornata una tupla dei territori coinvolti dopo l'avvenimento di esso). Si è deciso però di rendere disponibile anche il recupero dello stato intero della partita, esponendo il metodo *getPartita()*, per eventuali errori di comunicazioni mancate. Si è deciso anche di delegare alcune funzionalità a sotto-controller, in modo da poter spostare la logica di funzionalità complesse da *GiocoController* per un mantenimento più facile.

N.B. Per una leggibilità migliore, durante le sequenze non mostriamo che *GiocoController* chiama le funzionalità dei sotto-controller.

Diagramma di Dettaglio: Broker



Abbiamo poi scelto di usare il pattern Broker. Questo ci permette di disaccoppiare i vari sottosistemi.

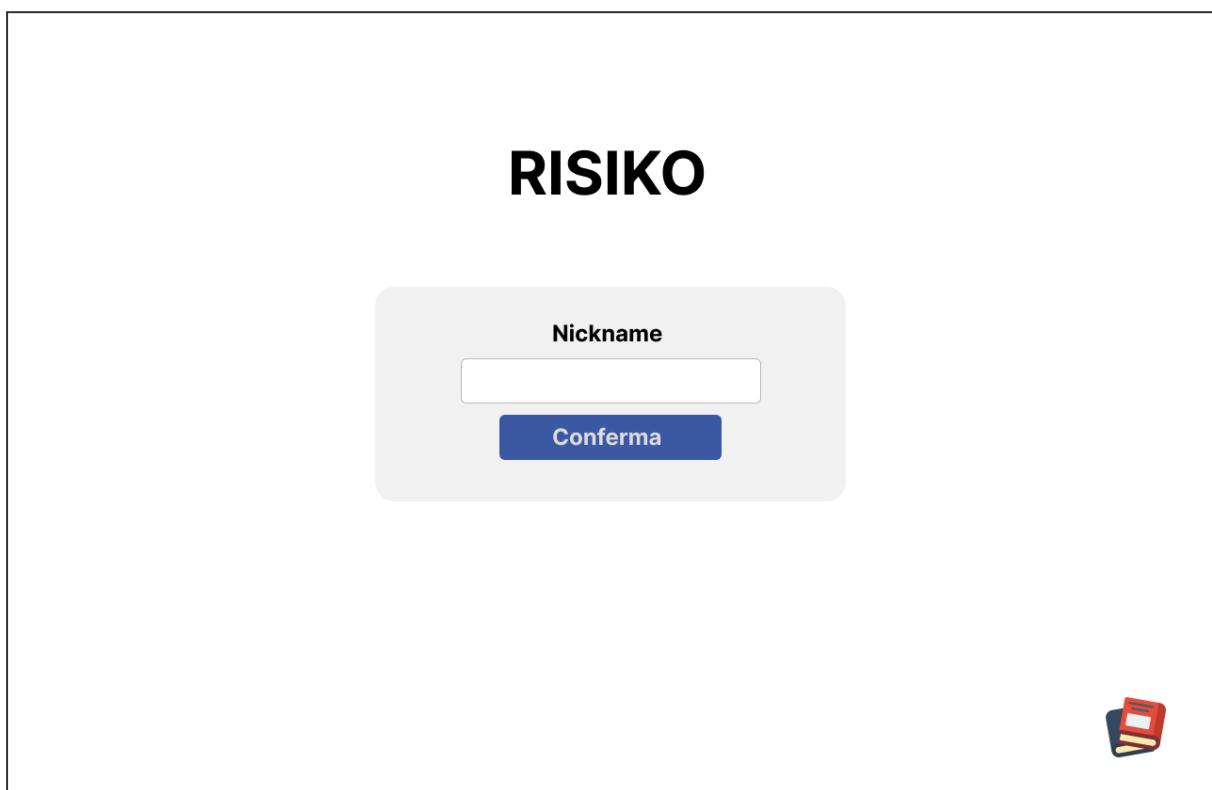
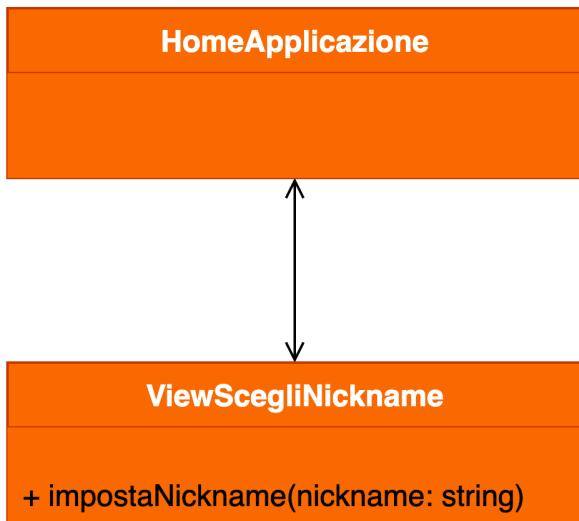
Nel nostro caso, siccome abbiamo scelto la tecnologia React, il Broker dovrà usare i Token (access e refresh) per autorizzare le richieste.

È stato deciso di lasciare il diagramma vuoto in quanto la sua realizzazione è lasciata alla discrezione dello sviluppatore.

Diagramma di Dettaglio: Client

I diagrammi seguenti rappresentano il Client. In particolare le classi rosse rappresentano le varie interfacce che ci saranno nella Build React (che viene mostrata più avanti nel documento nella Progettazione per il Deployment). Viene mostrata inoltre la relativa interfaccia grafica per ogni pagina e view.

Scegli nickname



Consulta manuale

ViewManuale

+ mostraRegole()

+ nascondiRegole()

RISIKO

SCOPO DEL GIOCO

Raggiungere per primi il proprio obiettivo segreto.

COMPONENTI

Un piano di gioco, rappresentante un planisfero suddiviso in 42 territori, appartenenti ai 6 continenti. Su ogni territorio è indicato un Punteggio Vittoria. Tale valore serve esclusivamente se decidete di utilizzare le nuove regole per il gioco a tempo ridotto.

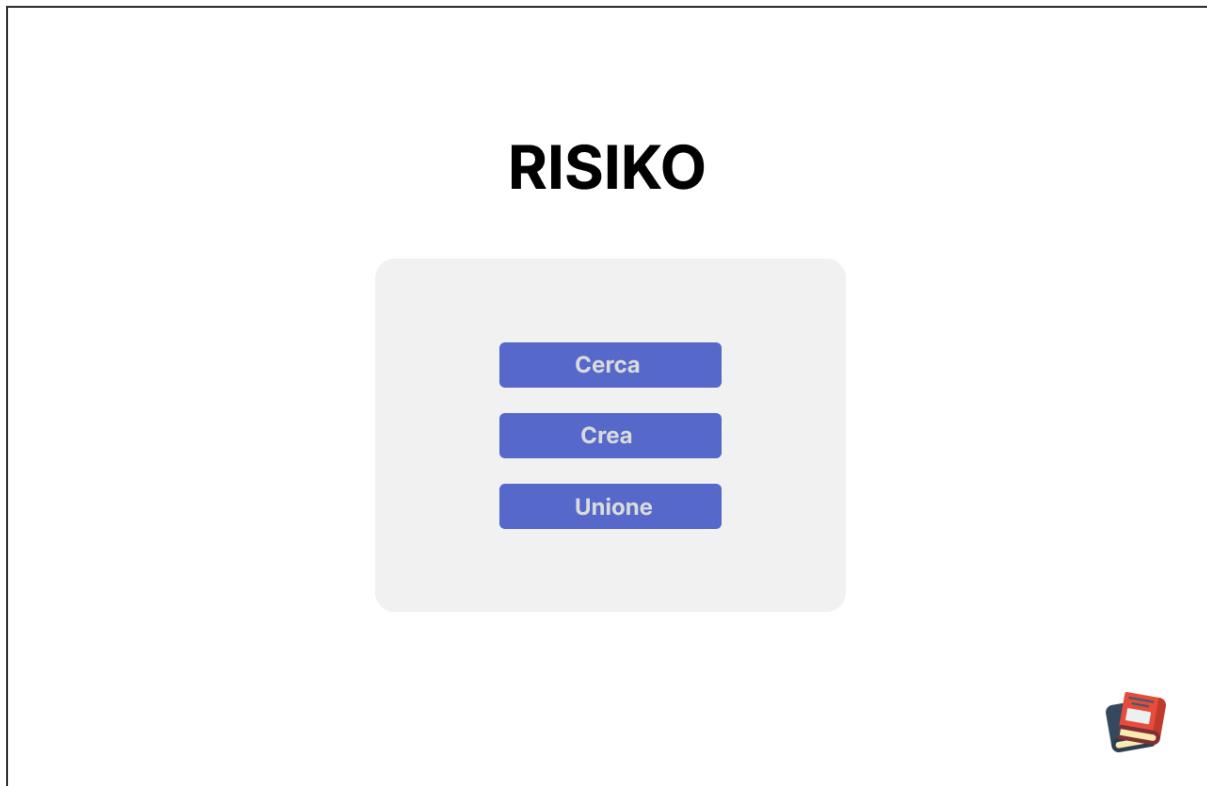
CONFERMA

Accesso gioco



Da notare come *ViewGestioneLobby* (view per un creatore di una partita all'interno di una Lobby) e *ViewLobby* (view per un giocatore che ha eseguito l'accesso a una Lobby tramite ricerca o unione) abbiano l'attributo `WebSocket`, che rappresenta la connessione con il server `WebSocket` associato alla Lobby in cui un giocatore si trova.

Home - View Accesso



View Cerca

Ricerca partita

Numero giocatori

Turni massimi

Mappa

Lingua chat

ANNULLA **CONFERMA**



View Crea

Crea partita

Numero giocatori

Turni massimi

Mappa

ANNULLA **CONFERMA**



[View Gestione Lobby](#)

Gestione Lobby

Giocatori

Arge	<button>ESPELLI</button>
Zen	<button>ESPELLI</button>
Pelle	<button>ESPELLI</button>

[SCIOLGI LOBBY](#) [AVVIA PARTITA](#)



[View Unione](#)

Unione partita

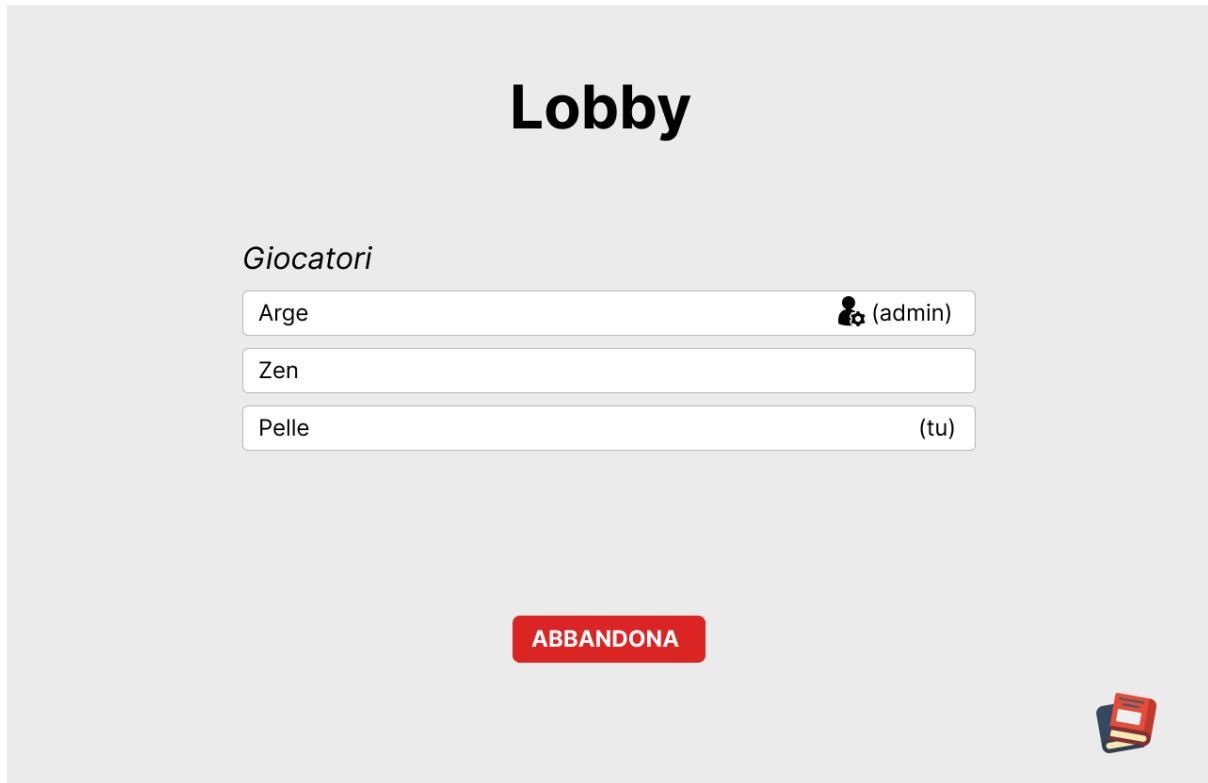
Codice partita

[ANNULLA](#)

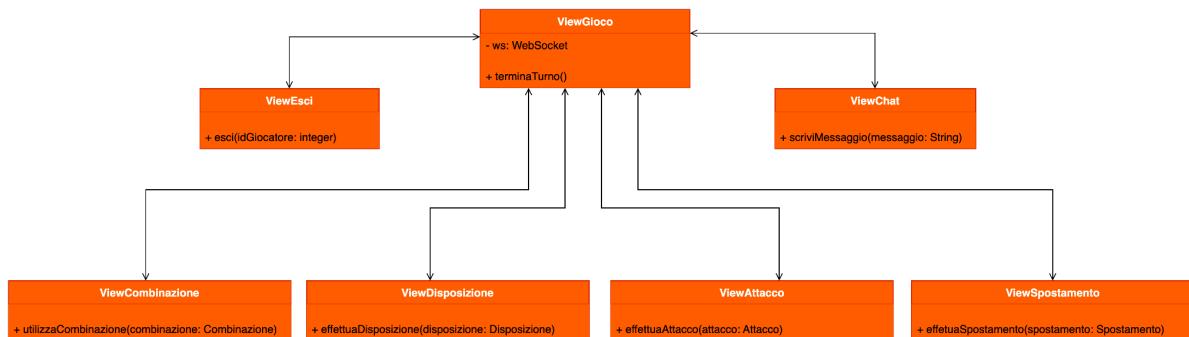
[CONFERMA](#)



View Lobby



Gestione partita



Da notare come *ViewGioco* abbia l'attributo *WebSocket*, che rappresenta la connessione con il server *WebSocket* associato alla *PartitaCominciata* in cui un giocatore si trova.

View gioco

RISIKO

Attesa turno di Pelle...

ABANDONA

Azioni di Pelle

Combinazione
Pelle: +7 armate

Dispersione
Pelle ha disposto 3 armate in Cina
Pelle ha disposto 4 armate in Italia

Mostra obiettivo

Mostra carte combinazione

Chat partita

Zen
Messaggio 12:35

Argo 10 T / 4 C
Zen
Pelle

T: numero territori
C: carte territorio

View combinazione

RISIKO

Fase di Combinazione

ABANDONA

Azioni svolte

Combinazione
Pelle: +7 armate

Dispersione
Pelle ha disposto 3 armate in Cina
Pelle ha disposto 4 armate in Italia

Mostra obiettivo

Mostra carte combinazione

Scegli Combinazione

Passa alla fase di disposizione

Tempo rimanente
2:12

Chat partita

Zen
Messaggio 12:35

Argo 10 T / 4 C
Zen
Pelle

T: numero territori
C: carte territorio

RISIKO

Fase di Combinazione

[ABANDONA](#)

Azioni svolte

Combinazione
Pelle: +7 armate

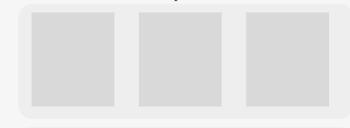
Dispersione
Pelle ha disposto 3 armate in Cina
Pelle ha disposto 4 armate in Italia

Scegli combinazione

Carte combinazione possedute



Combinazioni possibili



+7 armate

+7 armate

[ANNULLA](#)

[ESEGUI](#)

[combinazione](#)

[Passa a fase di dispersione](#)

Chat partita

Zen rimosso 12:35

Arge 10 T / 4 C

Pelle 0 territori



View disposizione

RISIKO

Fase di Disposizione

[ABANDONA](#)

Azioni svolte

Combinazione
Pelle: +7 armate

Dispersione
Pelle ha disposto 3 armate in Cina
Pelle ha disposto 4 armate in Italia



[Mostra obiettivo](#)

Armate da disporre

Tempo rimanente

2:12

[Mostra carte combinazione](#)

12

[Passa alla fase di attacco](#)

Chat partita

Zen Messaggio 12:35

Arge 10 T / 4 C

Zen

Pelle

T: numero territori
C: carte territorio



View attacco

RISIKO

Fase di Attacco

ABANDONA

Azioni svolte

Combinazione
Pelle: +7 armate

Dispersione
Pelle ha disposto 3 armate in Cina
Pelle ha disposto 4 armate in Italia

BRASILE
Selezione il territorio dal quale avviare l'attacco
SELEZIONA

Mostra obiettivo **Mostra carte combinazione** **Passa alla fase di spostamento**

Tempo rimanente
2:12

Chat partita

Zen
Messaggio 12:35

Argo 10 T / 4 C
Zen
Pelle

T: numero territori
C: carte territorio

RISIKO

Fase di Attacco

ABANDONA

Azioni svolte

Combinazione
Pelle: +7 armate

Dispersione
Pelle ha disposto 3 armate in Cina
Pelle ha disposto 4 armate in Italia

VENEZUELA
Scegli il numero di armate con cui attaccare

ATTACCA

Mostra obiettivo **Mostra carte combinazione** **Passa alla fase di spostamento**

Tempo rimanente
2:12

Chat partita

Zen
Messaggio 12:35

Argo 10 T / 4 C
Zen
Pelle

T: numero territori
C: carte territorio

RISIKO

Fase di Attacco di Zen

ABANDONA

Azioni svolte

Combinazione

Pelle: +7 armate

Dispersione

Pelle ha disposto 3 armate in Cina

Pelle ha disposto 4 armate in Italia



Mostra obiettivo

Tempo rimanente

0:45

Mostra carte
combinazione

Chat partita

Zen

Messaggio

12:35

>

● Arge 10 T / 4 C

● Zen

● Pelle

T: numero territori
C: carte territorio



View spostamento

RISIKO

Fase di Spostamento

ABANDONA

Azioni svolte

Combinazione

Pelle: +7 armate

Dispersione

Pelle ha disposto 3 armate in Cina

Pelle ha disposto 4 armate in Italia



Mostra obiettivo

Tempo rimanente

2:12

Mostra carte
combinazione

Termina turno

Chat partita

Zen

Messaggio

12:35

>

● Arge 10 T / 4 C

● Zen

● Pelle

T: numero territori
C: carte territorio



RISIKO

Fase di Spostamento

ABANDONA

Azioni svolte

Combinazione

Pelle: +7 armate

Dispersione

Pelle ha disposto 3 armate in Cina

Pelle ha disposto 4 armate in Italia



Chat partita

Zen

Messaggio

12:35

>

● Arge 10 T / 4 C

● Zen

● Pelle

T: numero territori
C: carte territorio



Mostra obiettivo

Tempo rimanente

2:12

Termina turno

View esci

RISIKO

Fase di Combinazione

ABANDONA

Azioni svolte

Combinazione

Pelle: +7 armate

Dispersione

Pelle ha disposto 3 armate in Cina

Pelle ha disposto 4 armate in Italia



Chat partita

Zen

Commento rimosso

12:35

>

● Arge 10 T / 4 C

● Zen

● Pelle

T: numero territori
C: carte territorio



Mostra obiettivo

ESCI

Esci dalla partita

Sei sicuro di voler uscire?

ANNULLA

ESCI

Mostra obiettivo

Mostra carte combinazione

Passa a fase di disposizione

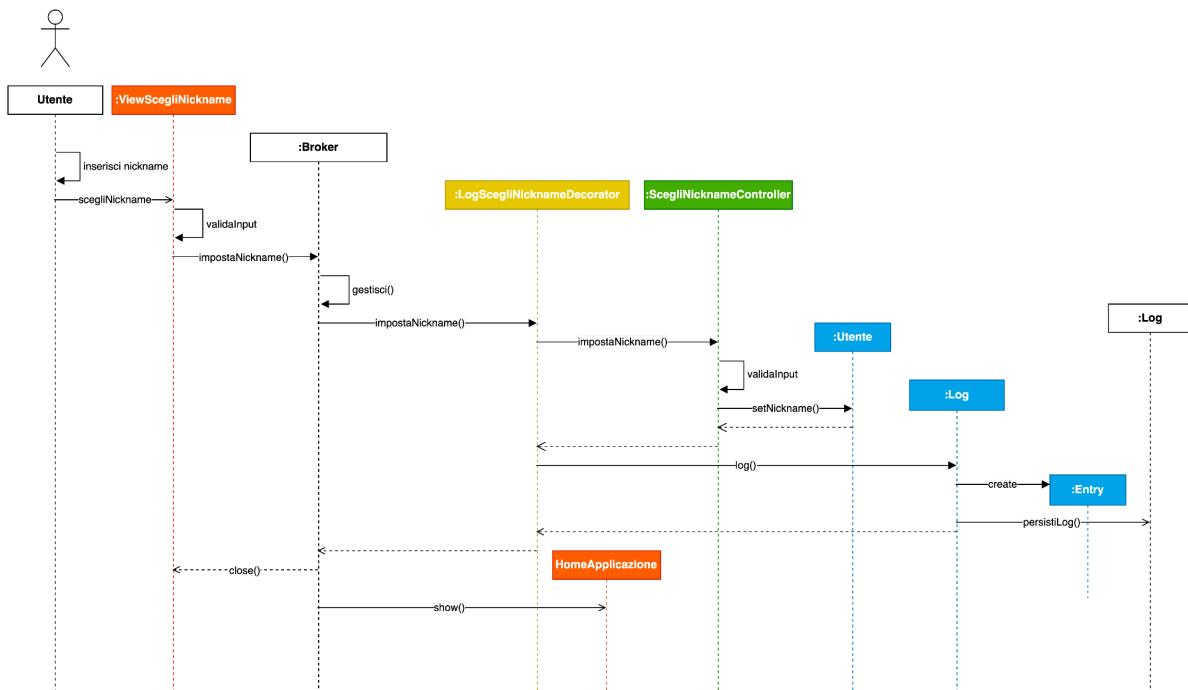
React component

Come già detto, il client dell'applicazione verrà sviluppato utilizzando React, quindi è bene ricordare che, anche se non è specificato nei diagrammi precedenti per motivi di leggibilità, ogni view estende *React.Component* e che quindi ha un proprio stato e della propria logica di visualizzazione, definita con un override del metodo *render()*.



Interazione

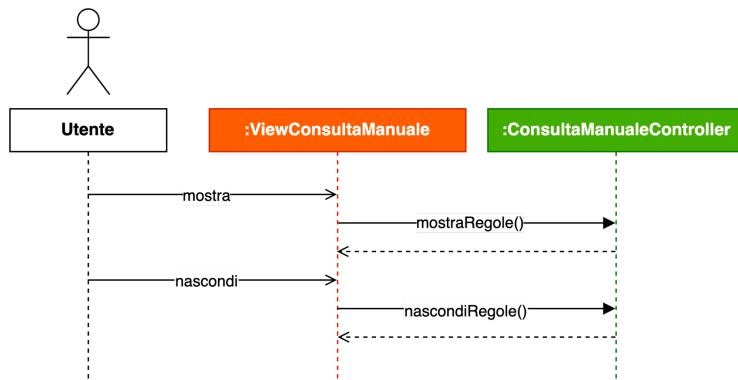
Diagramma di Sequenza: Impostazione del nickname



Si noti che in progettazione la validazione dell'input avviene sia lato client che lato server, per prevenire input errati e attacchi di tipo *Code injection*.

In questa sequenza mostriamo anche i passaggi che avvengono per scrittura sul Log fisico. Quest'ultimo non verrà riportato con questo livello di dettaglio nelle prossime sequenze. La scrittura su Log è asincrona non bloccante, in modo da non creare ulteriore ritardo di risposta per i vari Client.

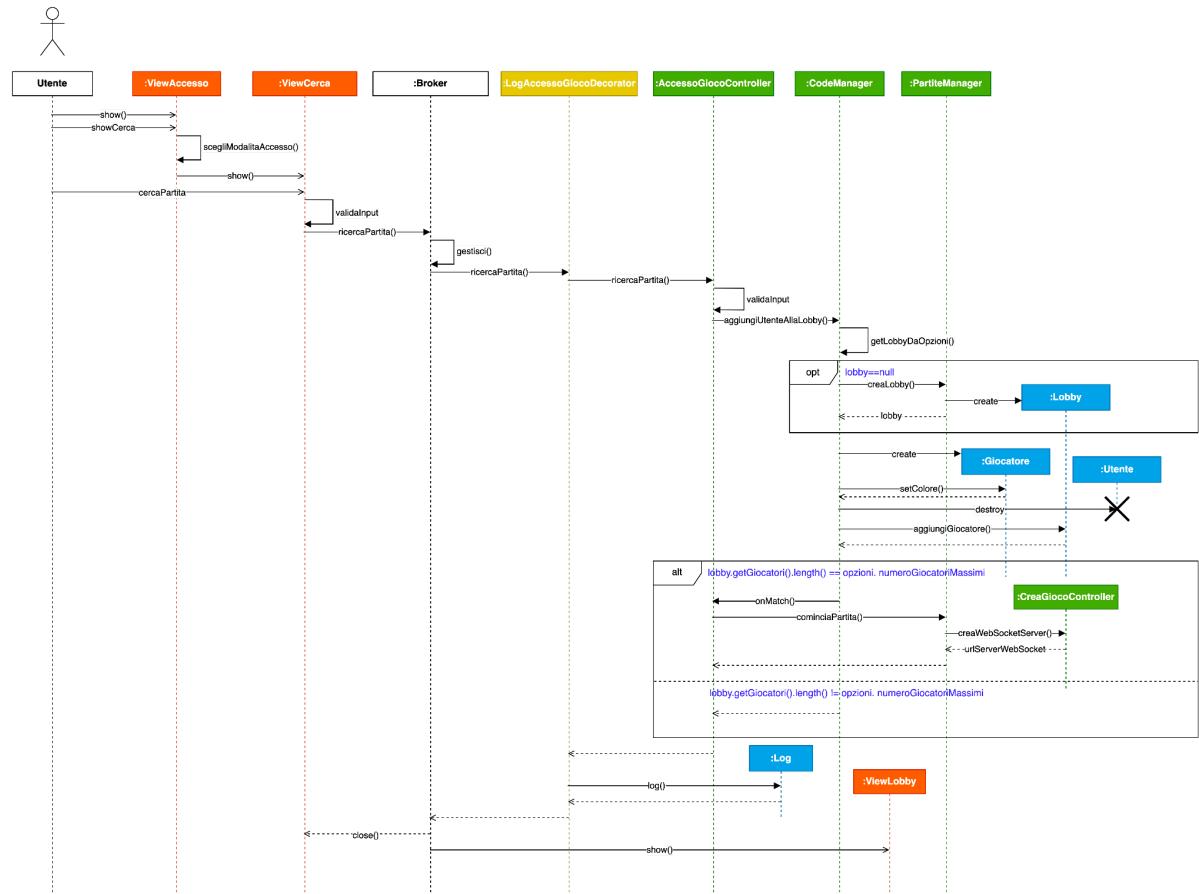
Diagramma di Sequenza: Visualizza manuale



Il diagramma mostra il procedimento di consultazione del manuale. Viene mostrato come l'utente, in qualsiasi momento del gioco, possa consultare il manuale e chiuderlo una volta terminata la lettura delle parti intere.

Da notare come la `ViewConsultaManuale` sia sempre visibile (icona del manuale in basso a destra nelle interfacce mostrate) e che si espande al click, per ridursi di nuovo a icona quando l'utente decide di nascondere le regole.

Diagramma di Sequenza: Accesso ad una partita tramite Ricerca



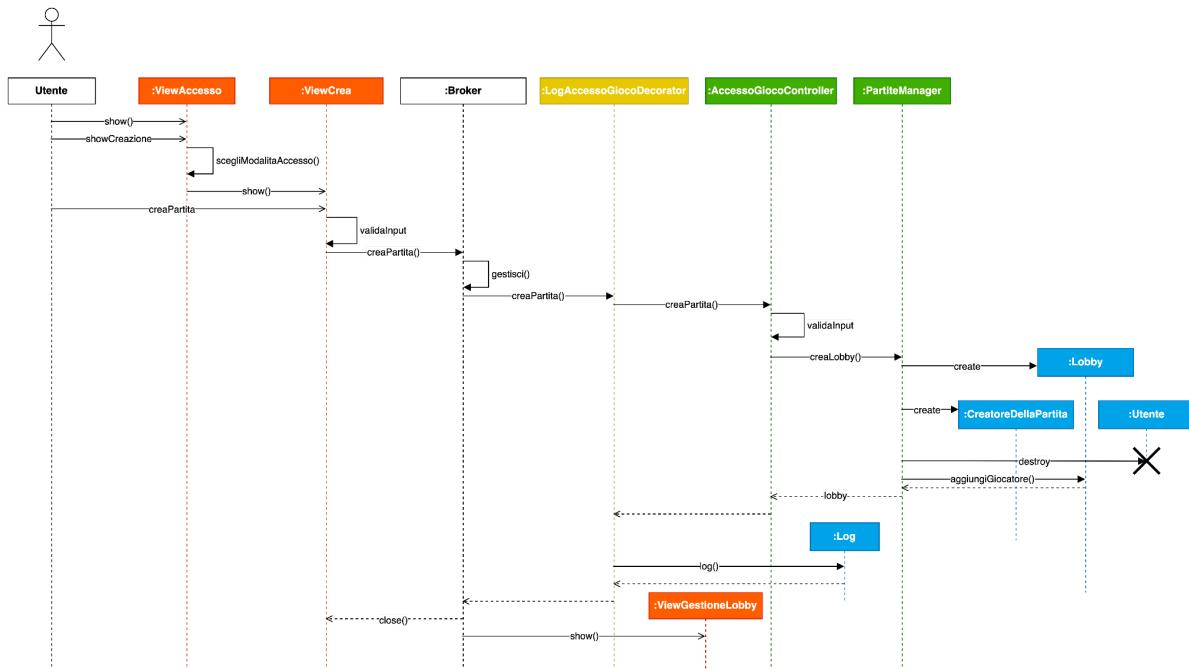
NB: Come fatto in analisi in questa sequenza e in quelle di creazione ed unione a una partita avviene anche l'assegnazione del colore. Mostriamo solo in questa sequenza tale operazione per motivi di spazio.

Si noti che il parsing dei parametri viene eseguito prima della chiamata del servizio del controller, quindi a differenza dell'analisi non viene mostrato che vengono creati, ma estratti dalla richiesta. Questo vale anche per le sequenze a seguire.

Inoltre il setting dei parametri non viene eseguito perché le proprietà vengono passate al costruttore (esempio opzioni alla Lobby).

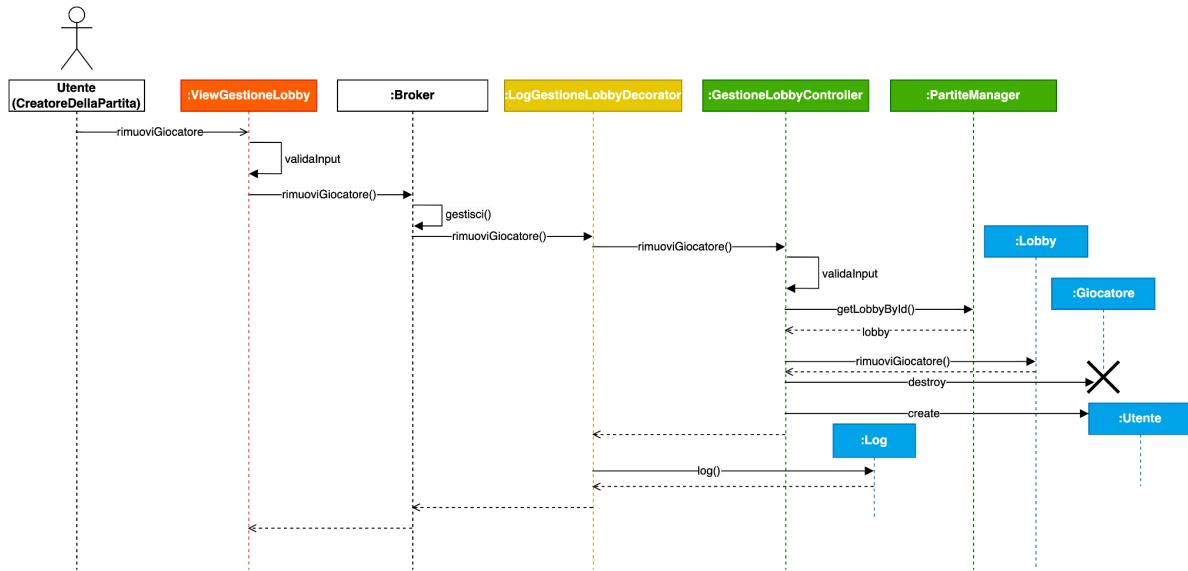
Da notare che `creaWebSocketServer()` viene chiamato tramite richiesta http, poiché `CreaGiocoController` si trova in un altro server.

Diagramma di Sequenza: Accesso ad una partita tramite Creazione



In questo diagramma è importante sottolineare che `aggiungiGiocatore()` esegue un controllo sul giocatore passato come parametro: se è un'istanza di `CreatoreDellaPartita` viene chiamato `setCreatore()`.

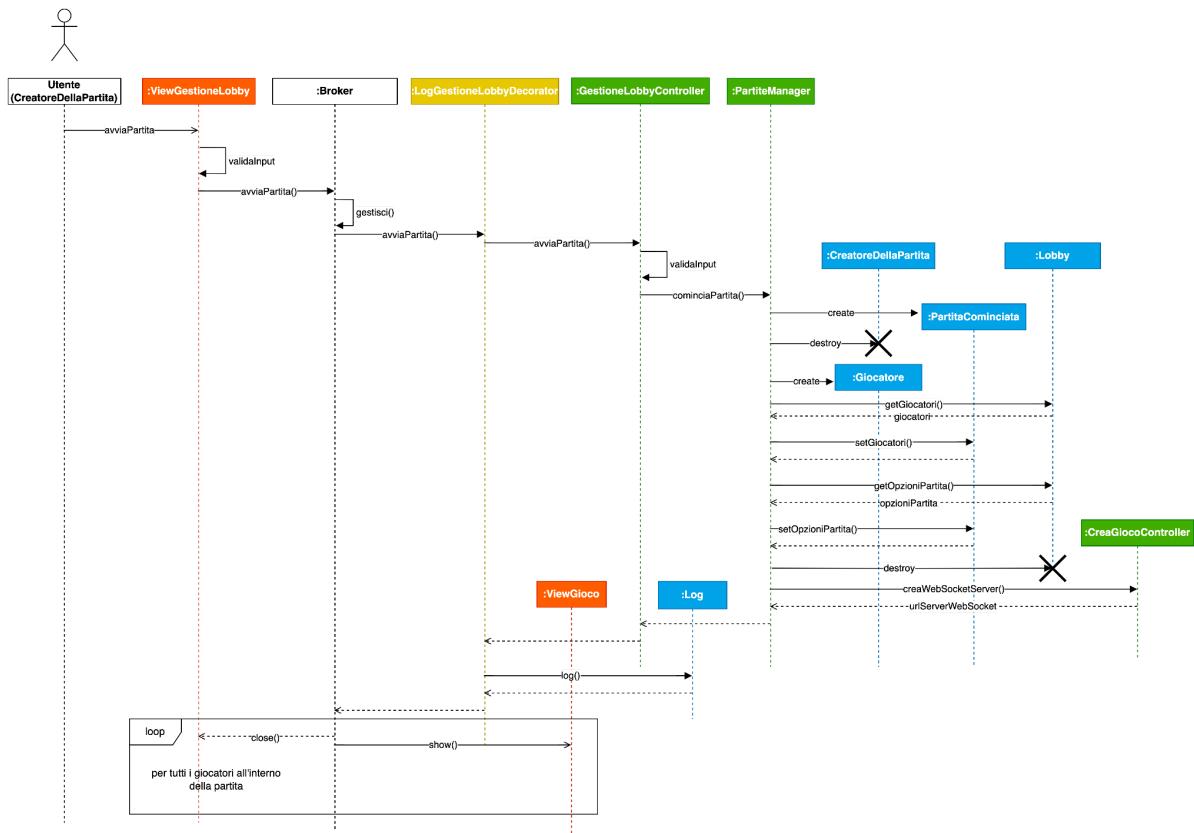
Diagramma di Sequenza: Espulsione Giocatore



Molti metodi di utilità come `getLobbyById()` in questa sequenza e nelle prossime non vengono mostrati per motivi di leggibilità. Mostriamo un esempio che però non verrà ripetuto nelle prossime sequenze.

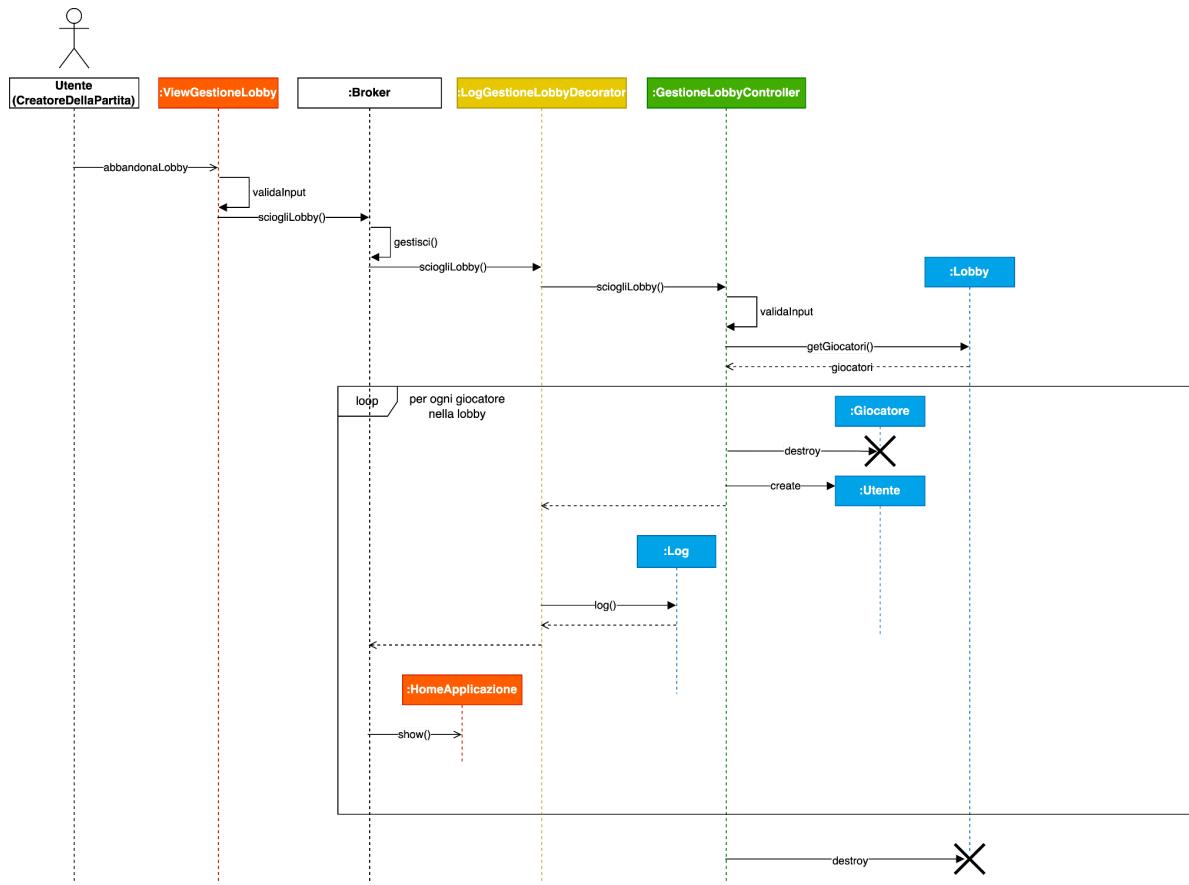
In questa sequenza è importante notare che i metodi del controller sono invocati in seguito a richieste HTTP, dopodiché nell'invocazione del servizio viene recuperato il server WebSocket associato alla lobby e viene mandato un messaggio notifica ai partecipanti. Tale comportamento vale anche per le prossime due sequenze.

Diagramma di Sequenza: Avviamento Partita



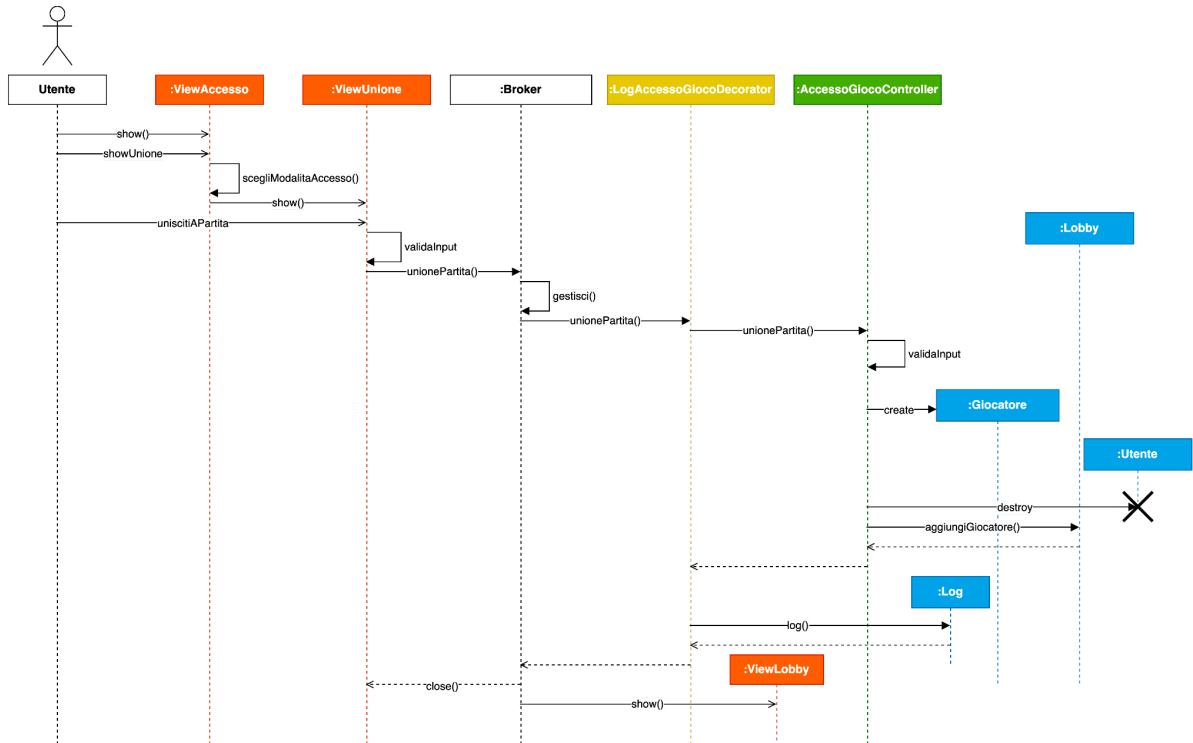
In questo diagramma mostriamo nel dettaglio il metodo `cominciaPartita()`, già visto in precedenza.

Diagramma di Sequenza: Scioglimento Lobby



In questo diagramma non ci sono punti critici su cui c'è bisogno di porre più attenzione.

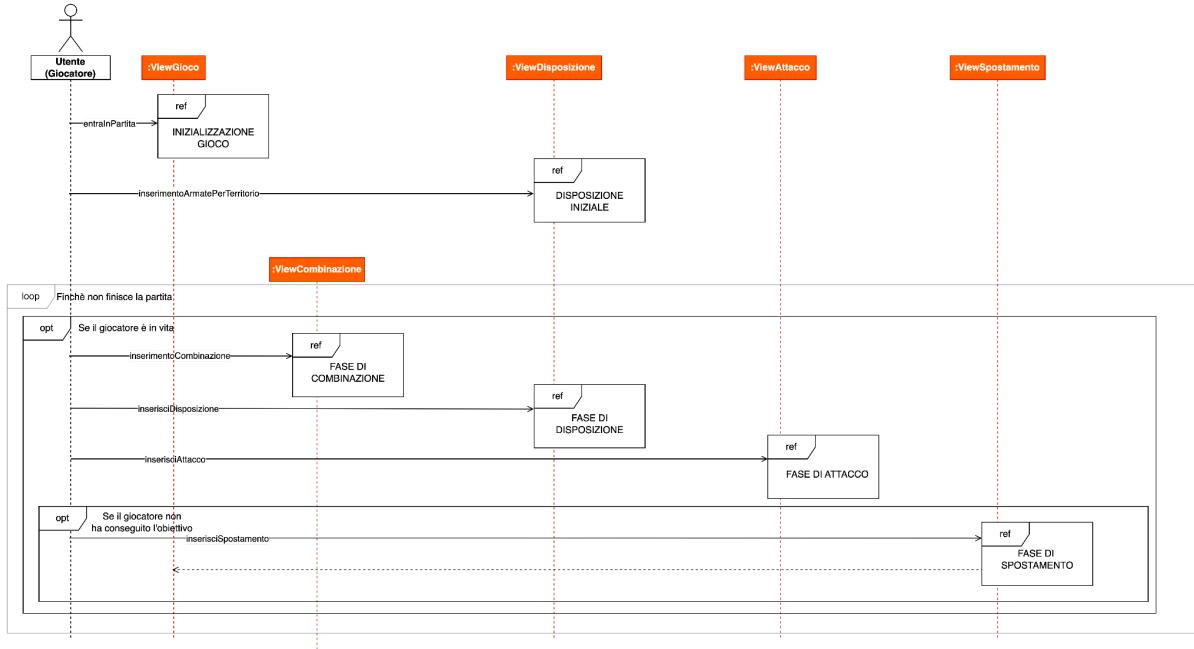
Diagramma di Sequenza: Accesso ad una partita tramite Unione



Si noti che in questo diagramma, la sequenza mostra come un utente possa unirsi a una Lobby. Ovviamente non viene ripetuto il fatto che si utilizza la classe *PartiteManager* per recuperare la Lobby relativa.

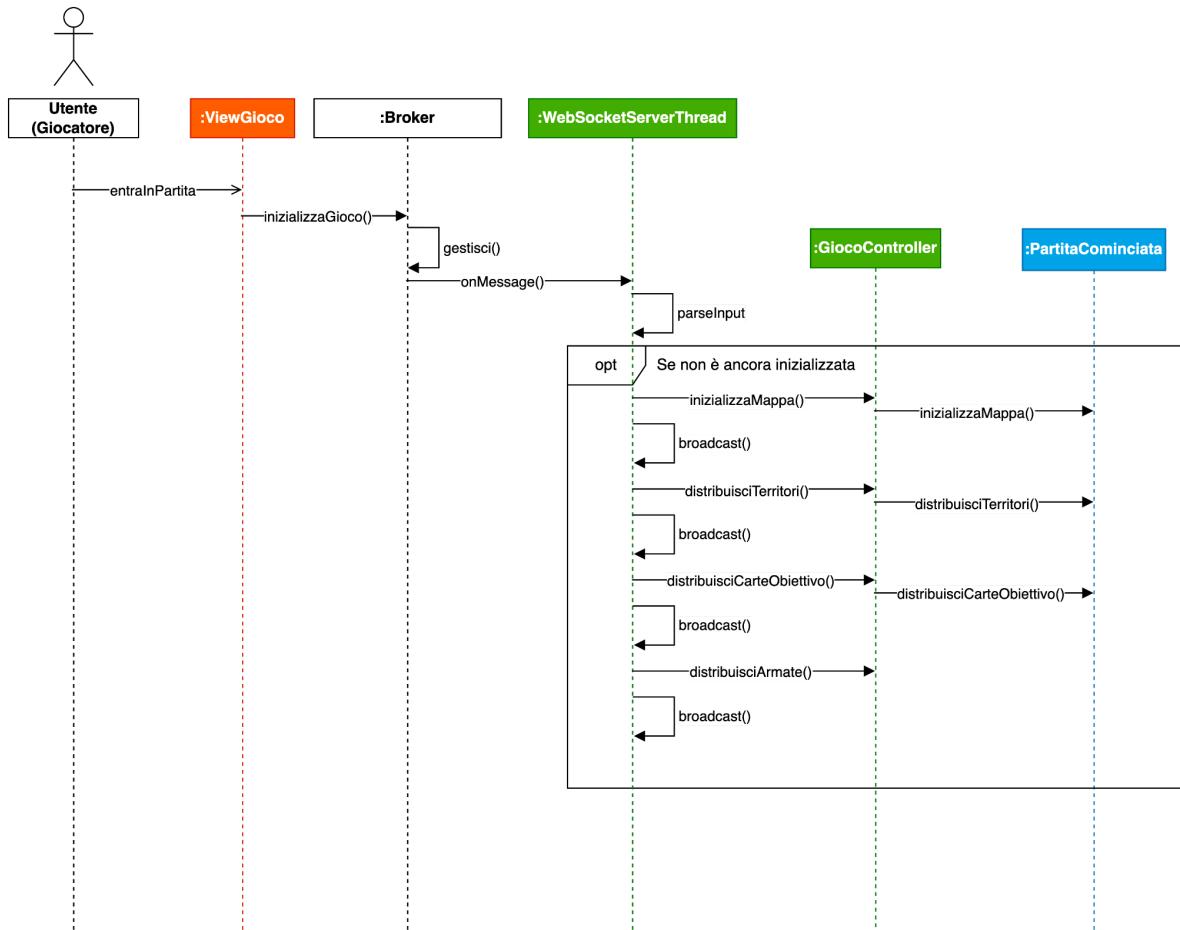
Diagramma di Sequenza: Gioca Partita

I diagrammi che seguono, sono affetti dello stesso principio descritto in Analisi, ovvero si utilizzano componenti per motivi di spazio. Inoltre per le sequenze a seguire l'utilizzo dei decorator verrà mostrato meglio nel dettaglio nella sequenza di combinazione e verrà omessa nelle altre.



Questo diagramma mostra la sequenza di un'intera partita.

Inizializza partita



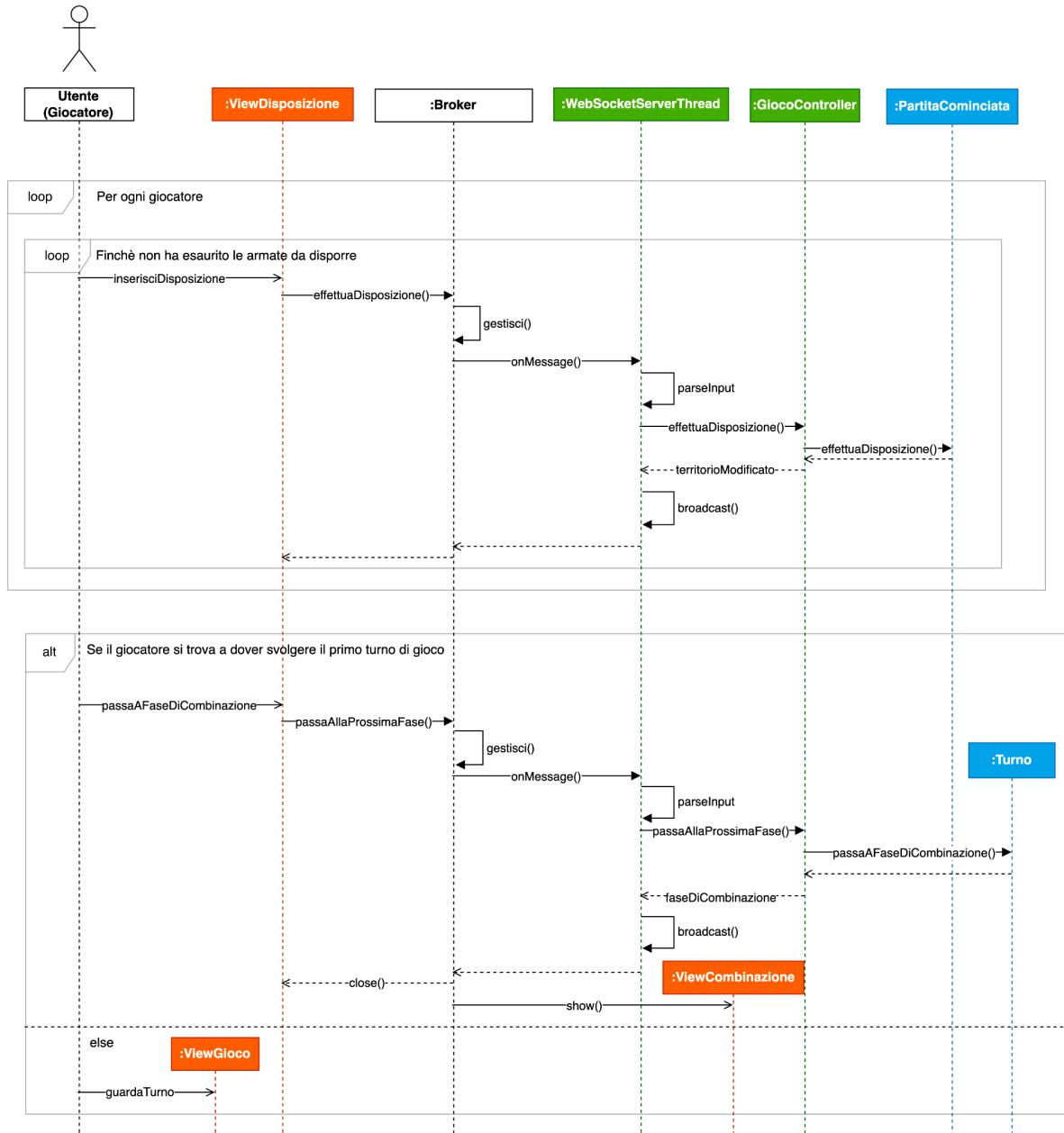
In questa fase, non si evidenziano i ritorni dei metodi sincroni.

In questa sequenza è importante notare l'utilità di *inizializzaMappa()*, che si occupa di caricare dinamicamente le caratteristiche della mappa, quali Regioni, Territori, confinane tra Territori e altri metadati. Questi, infatti, non vengono caricati nella creazione delle opzioni della Lobby corrispondente, per ottimizzare i tempi di attesa.

Si noti anche che per semplicità non vengono mostrati i dettagli di *distribisciTerritori()* e *distribisciCarteObiettivo()*, essendo metodo con algoritmi complessi. Questi ultimi fanno uso della classe *MazzoDiGioco* citata nel dominio.

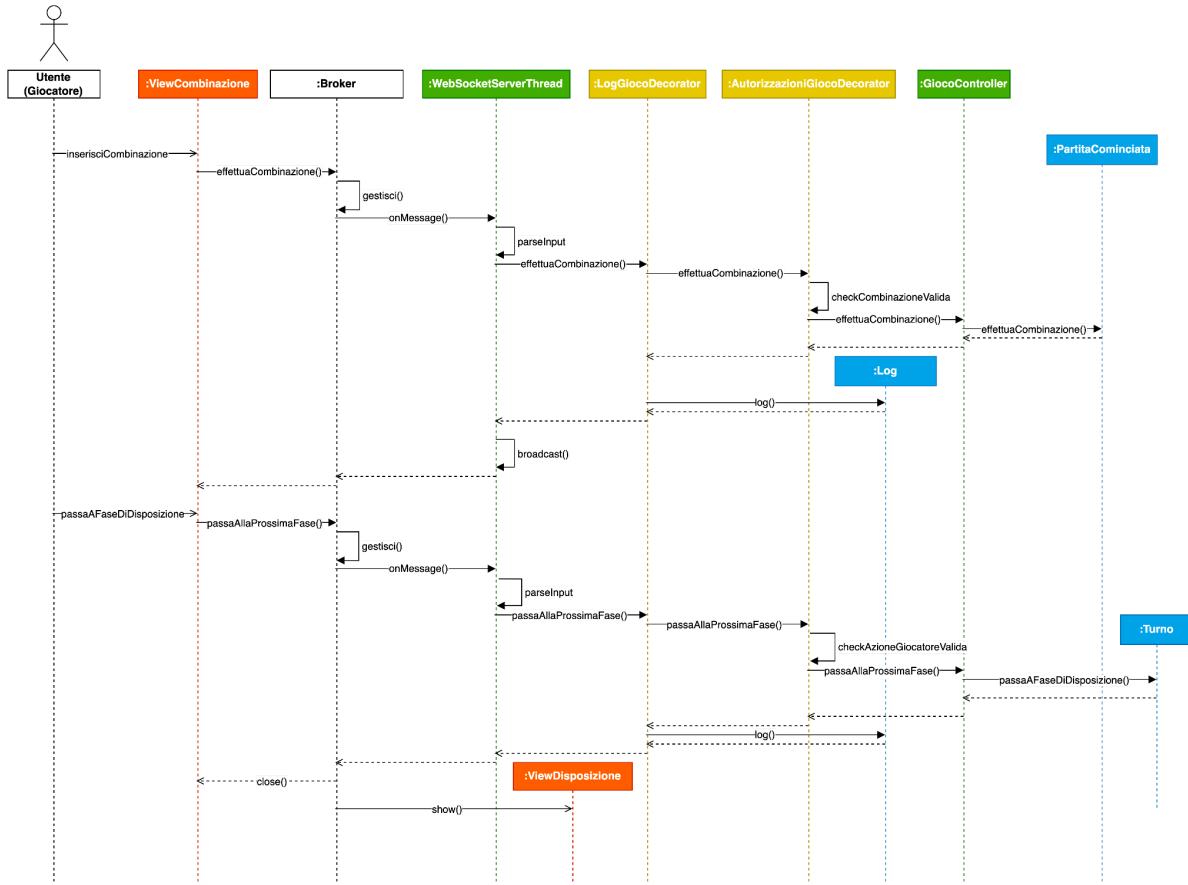
Da notare che *distribisciArmate()* distribuisce le armate effettivamente utilizzabili, siccome il metodo *distribisciTerritori()* effettua la distribuzione assegnando già un'armata a ogni territorio.

Disposizione iniziale



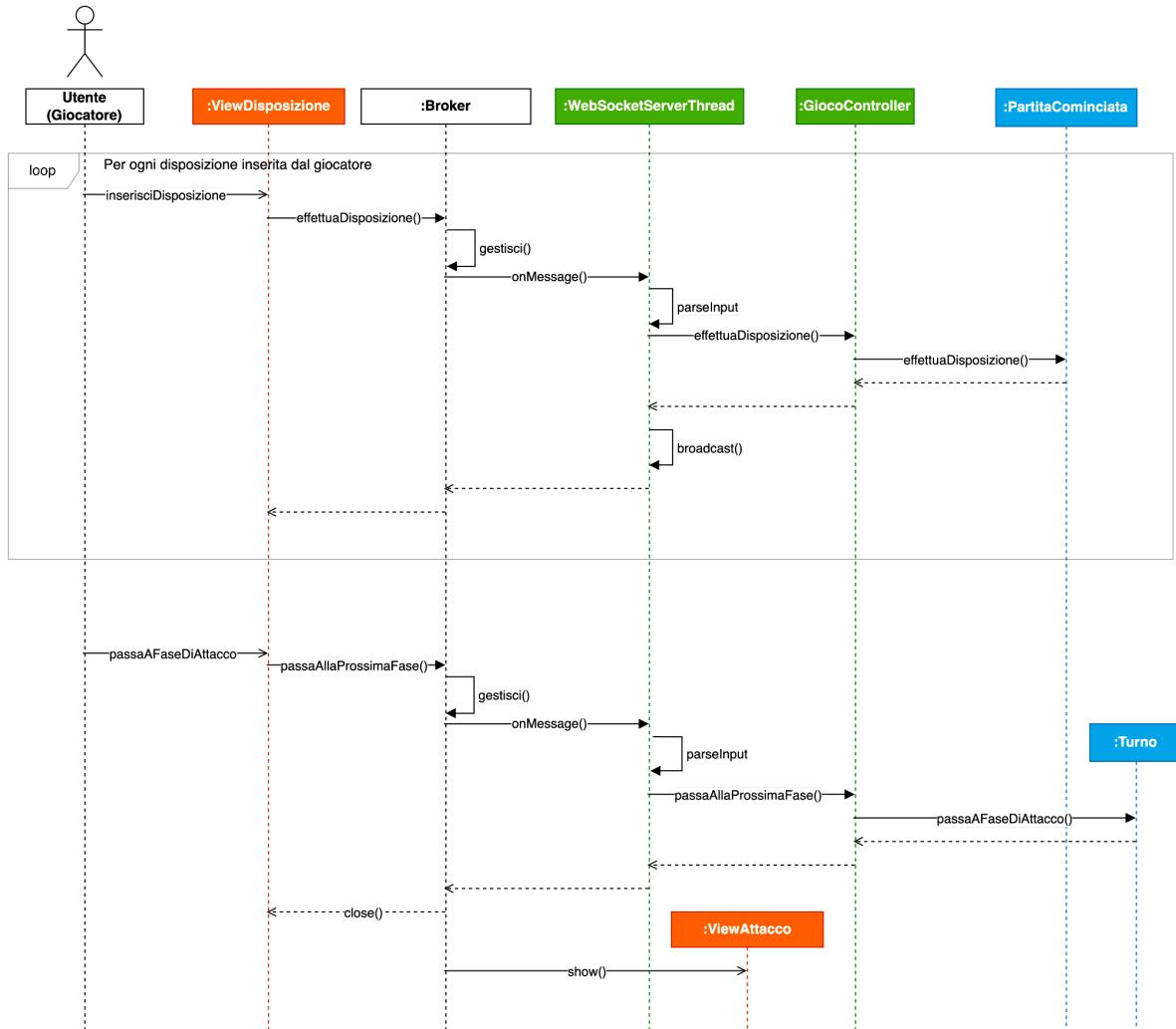
Da notare che i parametri vengono estratti dal JSON ricevuto prima di richiamare la funzionalità del controller, facendo il parsing e costruendo i modelli del dominio.
Il Turno viene recuperato con `_gioco.getPartita().getTurnoCorrente()`.

Fase di combinazione



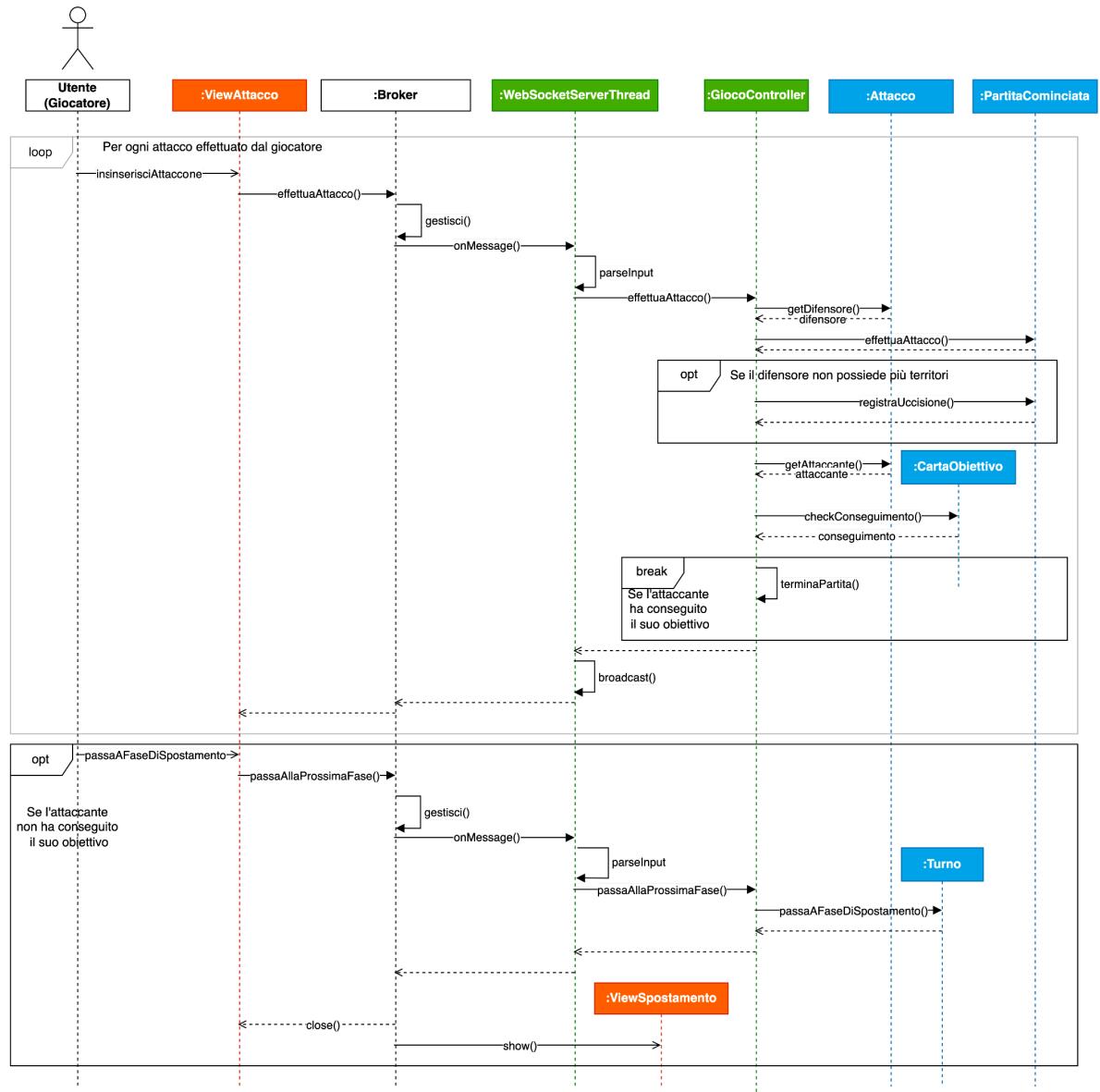
Come mostrato già in sequenze precedenti alla fine di ogni azione avviene un *broadcast()*. Per motivi di spazio, non mostriamo che *AutorizzazioniGiocoDecorator*, ad ogni check, se il controllo restituisce risultato negativo, il forward della chiamata non avviene.
Si noti che, anche se non viene mostrato, quando si utilizza una combinazione vengono rimescolate le carte coinvolte nel *MazzoDiGioco*.

Fase di disposizione



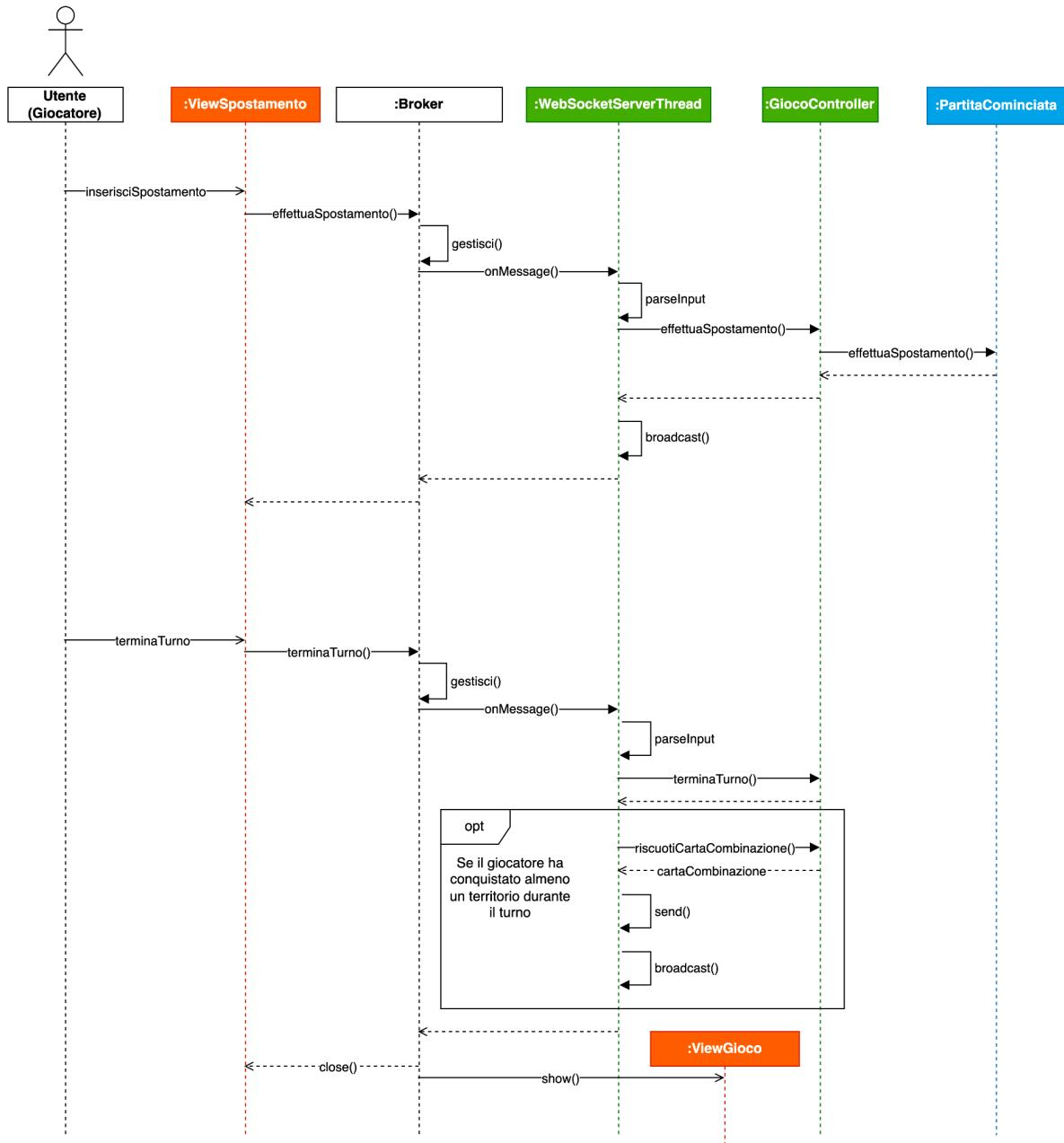
In questa sequenza non ci sono particolari punti da evidenziare.

Fase di attacco



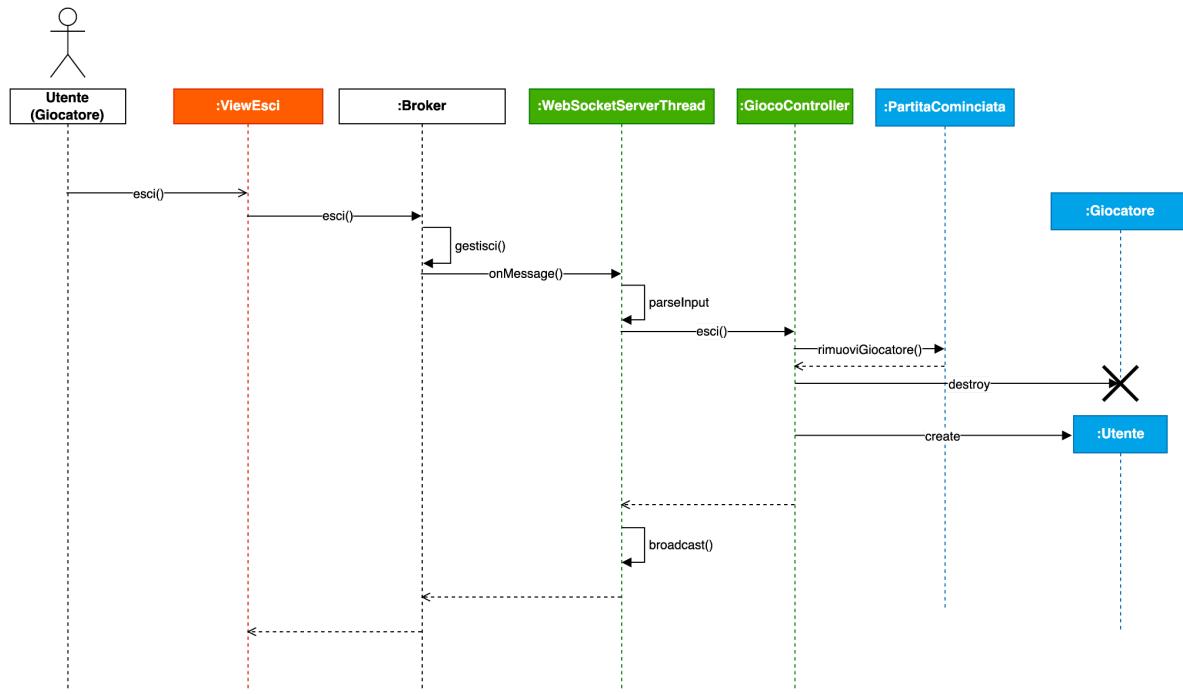
Per motivi di spazio non vengono mostrate, però le sequenze di `getDifensore()` e `getAttaccante()` sono le stesse mostrate in analisi.

Fase di spostamento



Da notare che se il Giocatore ottiene una carta combinazione (sempre utilizzando la classe *MazzoDiGioco*), la *send()* condivide la carta combinazione al giocatore interessato e il metodo *broadcast()* notifica che quel giocatore ha ottenuto una carta combinazione.

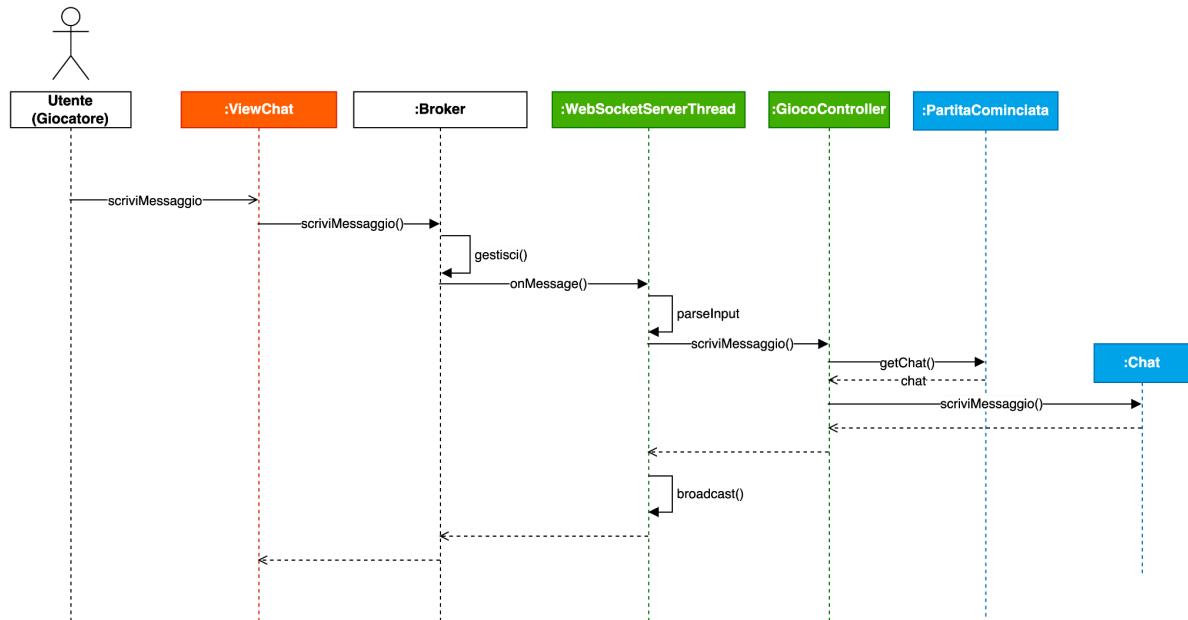
Diagramma di Sequenza: Esci dalla partita



L'uscita della partita è consentita in ogni momento della partita.

Da notare che la redistribuzione avviene nella chiamata di *rimuoviGiocatore()*;

Diagramma di Sequenza: Utilizzo Chat



In questo diagramma è importante commentare come lo scambio di messaggi non viene affatto da operazioni di log. Il sistema si occupa solo di sanificare l'input.

Comportamento

Non sono stati sviluppati altri diagrammi di stato, si utilizzano unicamente quelli presenti in Analisi.

Progettazione della persistenza

Formato File Log

- Formato file per Log delle operazioni:
DataOra | Tag | Messaggio
 - ‘Tag’ identifica quale sezione sta provocando la scrittura sul Log.
 - ‘Messaggio’ raccoglie tutte le operazioni che sono state eseguite, per informare in maniera dettagliata caso per caso.

Progettazione per il collaudo

Vanno completati i test per tutte le classi.

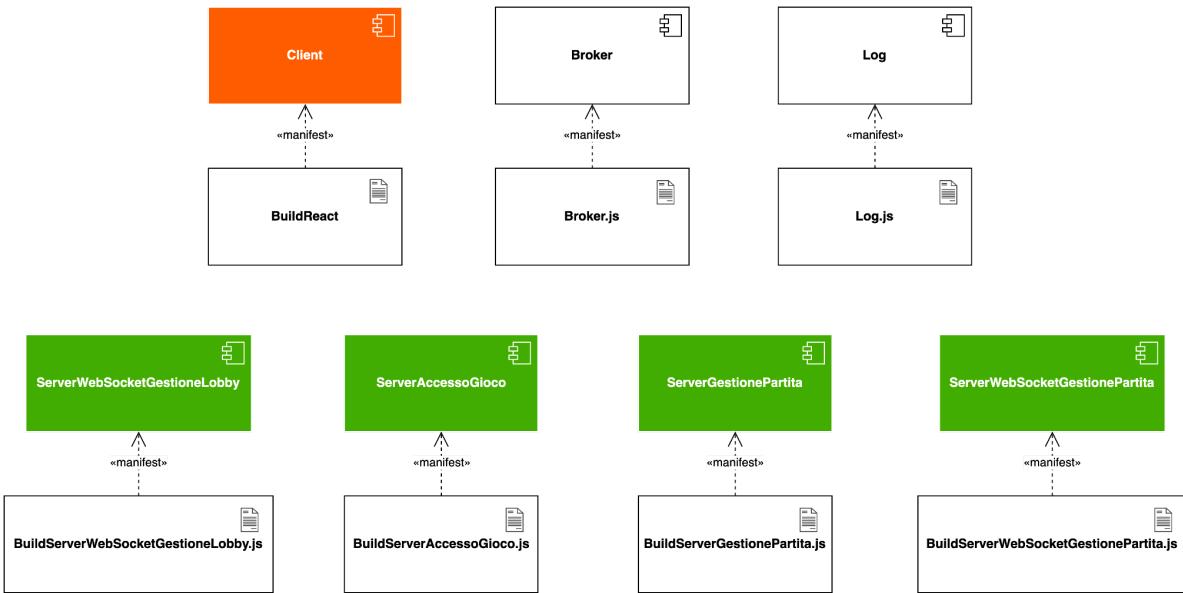
Progettazione per il Deployment

Verrà eseguito il deploy della build React (Client) su un server che servirà da Host (non dovendo installare il client sulle macchine dei vari client).

Per quanto riguarda il lato server è stato deciso di eseguire il deploy su due macchine server per le funzionalità di AccessoGioco e GestionePartita. Il server di AccessoGioco conterrà il server http per quanto riguarda le varie modalità di accesso e i vari server websocket relativi alle Lobby. Il server di GestionePartita, invece, conterrà il server http di creazione delle partite e i vari server websocket associati alle partite in corso. Sarà presente inoltre un server di Log. I server dovranno essere installati su macchine all'interno di una rete privata opportunamente protetta da un firewall a cripitura di pacchetti. L'unico punto di contatto verso l'esterno è il Broker.

Deployment

Artefatti



Deployment Type-Level

